Linguistically–driven Selection of Correct Arcs for Dependency Parsing

Felice Dell'Orletta, Giulia Venturi, and Simonetta Montemagni

Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC-CNR),
Italia**NLP** Lab – *www.italianlp.it*
Pisa, Italy

{felice.dellorletta,simonetta.montemagni,giulia.venturi}@ilc.cnr.it

Abstract. LISCA is an unsupervised algorithm aimed at assigning a quality score to each arc generated by a dependency parser in order to produce a decreasing ranking of arcs from correct to incorrect ones. LISCA exploits statistics about a set of linguistically–motivated and dependency–based features extracted from a large corpus of automatically parsed sentences and uses them to assign a quality score to each arc of a parsed sentence belonging to the same domain of the automatically parsed corpus. LISCA has been successfully tested on two datasets belonging to two different domains and in all experiments it turned out to outperform different baselines, thus showing to be able to reliably detect correct arcs also representing domain–specific peculiarities.

Keywords. Dependency parsing, correct arcs.

Selección de los arcos correctos basada en información lingüística para análisis sintáctico de dependencias

Resumen. LISCA es un algoritmo no supervisado cuyo objetivo es asignar un puntaje cualitativo a cada arco generado por el analizador sintáctico de dependencias con el fin de producir un ranking decreciente de los arcos desde los correctos hasta los incorrectos. LISCA usa la estadística del conjunto de características basadas en la información lingüística y dependencias que se extraen del corpus grande de frases analizadas sintácticamente por la computadora y las utiliza para asignar un puntaje cualitativo a cada arco de la frase analizada que pertenece al mismo dominio del corpus. LISCA se probó exitosamente utilizando dos conjuntos de datos de dos dominios distintos y en todos los experimentos su rendimiento fue mejor que el de varios métodos de referencia; así se demostró su capacidad de detectar los arcos correctos de manera confiable representando también las características específicas de los dominios.

Palabras clave. Análisis sintáctico de dependencias, arcos correctos.

1 Introduction

When applied to real–world texts (e.g. the web or domain–specific corpora such as bio–medical literature, legal texts, etc.), state–of–the–art statistical parsing systems have a significant drop of accuracy. However, if on the one hand creating a manually annotated training set represents the most safe way to improve the performance of a parser or to adapt it to a domain different with respect to the original training data, on the other hand this is a highly expensive task in terms of time and human effort. To overcome this problem, over the last few years a growing interest has been shown in exploring methods and techniques for assigning a plausibility score to the output of a parser. Depending on whether the scored output is represented by a full syntactic parse or by atomic syntactic constructions, these methods can be divided in two main research scenarios.

The first is the case, among others, of the work reported in [31, 12, 8, 20]. Automatically identified reliable parses can be usefully exploited in different applications and tasks, ranging from the domain adaptation task in a self–training scenario [24] to the minimization of human annotators' efforts in treebank construction [32].

In the second scenario, instead of identifying reliable parses the goal is the detection of correctly identified syntactic constructions: with dependency–based parsers, this coincides with the identification of correct arcs. This is the scenario we will be dealing with in this paper. Approaches proposed so far in the literature differ at the level of the specific task they were meant to support and the underlying methodology. Concerning the supported task, the studies devoted to the detection of correct arcs are typically carried out with a view to: *i)* improving the accuracy of statistical parsers, as proposed in [18, 3, 29, 2, 26]; *ii)* adapting statistical parsers to domain corpora outside of the data from which they were trained, as proposed in [4, 30]; *iii)* supporting the creation of hand–annotated syntactic resources (i.e. treebanks) by identifying problematic areas for human pre– and post–processing [1, 14, 13, 15].

From the methodological point of view, depending on whether sets of rules, training data or large amounts of automatically parsed data are exploited by the correct arc detection algorithm, approaches proposed so far can be classified into three different classes according to whether they are: *i)* rule–based, *ii)* supervised or *iii)* unsupervised. The first is the case of [13] and [15] who developed error detection methods based on a "gold" grammar, i.e. by comparing rules of the gold grammar with rules induced from automatically parsed data with the final aim of identifying anomalies in treebanks, or of [1] who used a combination of rule–based and statistical methods to detect treebank errors. [14] operate instead in a supervised scenario by using n–grams statistics extracted from "gold" data to detect variations in treebanks which could represent potential errors. [18], [3] and [2] developed a classifier to detect errors or to identify correct analyses in automatically parsed sentences: they compare correct parses of a treebank with incorrect trees automatically generated to produce the training corpus for the error classifier. Unsupervised approaches are adopted instead by [29], [30] and [26] who apply the point–wise mutual information score (or a function close to it) to specific lexico–syntactic configurations using statistics extracted from large automatically parsed data to compute the reliability of arcs for self–training purposes.

In this paper, we propose a new unsupervised approach to assign a quality score to individual arcs within the output of a dependency parser. The unsupervised nature of the method permits to avoid the development of sets of rules or the creation of manually annotated treebanks. Differently from other unsupervised approaches operating on automatically parsed data, the proposed approach is overtly devoted to the ranking of arcs (from correct to incorrect ones) whose evaluation is carried out in terms of the accuracy of ranking rather than of its effectiveness with respect to a specific task (e.g. the improvement of the parser accuracy). Other qualifying properties of our approach consist in its relying on linguistic features encoded in terms of dependency structures and in the definition of a new arc quality score. The paper is organised as follows: Section 2 describes our algorithm for the ranking of arcs; Sections 3 and 4 present respectively the baselines and the experimental set–up; achieved results are reported and discussed in Section 5.

2 The LISCA Algorithm

The LISCA (*LInguiStically–driven Selection of Correct Arcs*) algorithm takes as input a set of parsed sentences and it assigns to each dependency arc a score quantifying its reliability, where a dependency arc is defined as a triple (d, h, t), where d is the *dependent*, h is the syntactic *head* or *governor*, and t is the *type* of the dependency relation linking d to h. The assigned quality score is then used to rank dependency arcs by reliability. Note that, in principle, LISCA can operate on the output of whatever dependency parser.

LISCA operates in two different steps: 1) it collects statistics about a set of linguistically–motivated features extracted from a wide corpus of parsed sentences; 2) it calculates a quality score for each dependency link of a newly parsed sentence using the feature statistics extracted from the corpus used during step 1).

2.1 Selection of Features

The features underlying LISCA are all aimed at characterizing the arc being analyzed with respect to structural (both global and local) properties of the dependency tree including it. In particular, a first set of features is aimed at positioning the dependency arc within the tree, with respect to both its hierarchical structure and the linear ordering of words. This set of features based on the global tree structure is complemented by local features, represented respectively by dependency length, direction and plausibility. Selected features are said to be "linguistically–motivated" since they are based on the dependency tree structure, and in particular they are focused on structures widely agreed in the literature to reflect the syntactic and parsing complexity of sentences.

2.1.1 Locating a Dependency Arc within the Overall Tree Structure

This set of features is aimed at locating a given dependency link within the overall sentence structure. This is done by focusing on the dependent d of the arc t being considered. A first complex feature, aimed at locating t within the hierarchical tree structure, combines the distance of d from a) the root node, b) the closer leaf node, and c) the furthest leaf node. To this specific end, the dependency paths including t and going from the root node to the closer and further leaf nodes of d respectively are reconstructed. In both cases, the shortest dependency path is selected, whose length is measured by node counting. The length of these two dependency paths is used as a proxy for characterizing the positioning of the arc t within the dependency structure, in particular for reconstructing its depth of embedding.

This global feature is complemented by two features focusing on local dependency sub-trees and aimed at locating d with respect to the surface linear ordering of words. The first feature refers to the sub-tree headed by d and counts all its immediate dependents, which are partitioned into two classes according to whether they precede or follow the head at the level of linear ordering of words within the sentence: these two classes will be hereafter referred to as "pre-dependents" and "post-dependents" of d respectively. The second feature refers to the sister nodes of d which are reconstructed starting from the sub-tree governed by the head h of d: again, sister nodes of d are partitioned into two classes following their pre- or post-d ordering.

This set of features can be seen as the dependency–based counterpart of syntactic complexity measures such as node-counting algorithms that count the number of nodes in the phrase markers of syntactic constructions: this is the case of, e.g., local nonterminal count in [16] or the depth algorithm in [34], as well as of word–counting algorithms based on ratios involving the length of constituents in terms of words (see e.g. [19]).

2.1.2 Length and Direction of a Dependency Arc

This is a complex feature combining two different information types: namely, dependency length (henceforth, DL), i.e. the linear distance between the syntactic head h and the dependent d (computed in terms of intervening words), and dependency direction (henceforth, DD), used to distinguish between head–initial and head–final dependency arcs. For any dependency relation holding between the words w_i and w_j, if w_i is the head and w_j is the dependent, then the dependency length can be defined as the difference $i - j$. With this measure, adjacent words linked by a dependency link have an absolute DL of 1. When i is greater than j, DL is a positive number, meaning that the head occurs after the dependent. When i is smaller than j, DL is a negative number: this is the case of a head–initial dependency link. Whereas at the level of individual dependencies DD is a purely qualitative difference, when referred to a parsed corpus it represents a quantitative measure reflecting the relative proportion of head-initial and head–final dependencies in the collection of texts.

The measure of dependency length is widely acknowledged in the literature to reflect the structural complexity of a dependency structure. [21] and [17], for instance, claim that the syntactic complexity of sentences can be predicted with measures based on the length of dependency links, given the memory overhead imposed by very long distance dependencies. It can also be used to explain the mechanisms of children language learning [27]. On the parsing side, [10] proves that "the relative order of the words and the distance between them will also strongly influence the likelihood of one word modifying the other": more recently, [25] report that statistical parsers have a drop in accuracy when analyzing longer dependencies. Concerning DD, [33] and more recently [22] consider this measure useful for classifying languages typologically.

2.1.3 Dependency Arc Plausibility

Contrary to the previous ones focusing on structural properties of the tree, this feature is used to calculate the plausibility of a dependency arc given the part-of-speech of d and h as well as of the head father of h. This feature, which was first used in [12] for detecting reliable parses (named as "ArcPOSFeat"), turned out to be an effective feature also for the detection of correct individual dependency arcs.

2.2 Computation Score

The quality score of a dependency arc (henceforth, QS) results from the combination of the weights associated with the features sketched in Section 2.1. For the specific concerns of this study, QS is computed as a product of the individual feature weights. Therefore, for each dependency arc a within the set of dependency arcs A describing the whole set of input sentences, QS is as follows:

$$QS(A_i) = \prod_{y=1}^{n} Weight(A_i, f_y, S, C, r),$$

where A_i is the i–th dependency arc within A; n is the number of features being considered; S is the sentence including A_i as part of its dependency representation; C is the set of characteristics with respect to which the weight of individual features is computed; r refers to the span used for defining the sentence length range being considered; and $Weight(A_i, f_y, S, C, r)$ is the computed weight for the y–th feature.

Except for *ArcPOSFeat*, $Weight(A_i, f_y, S, C, r)$ is defined as:

$$Weight(A_i, f_y, S, C, r) = \prod_{\forall c \in C_{f_y}} \left(\frac{F(V(f_y), c, range(L(S), r))}{|range(L(S), r), c|}\right),$$

where $V(f_y)$ is the value of the y–th feature of A_i; $L(S)$ is the length of the sentence S; $range(L(S), r)$ defines the sentence length range covering values from $L(S) - r$ to $L(S) + r$; $F(V(f_y), range(L(S), r), c)$ is the frequency of $V(f_y)$ within the set of all dependency links sharing a given characteristic c with the arc A_i and occurring in sentences whose length is in the range $range(L(S), r$[1]); finally, $|range(L(S), r), c|$ is the number of arcs in A describing sentences whose length is in the range $range(L(S_i), r)$ and sharing the characteristic c.

For defining the weight associated with each feature, in LISCA the following characteristics c of d in A_i are taken into account: namely, the part–of–speech and the lemma. Whenever c is assigned the *None* value, $|range(L(S), r), None|$ thus corresponds to the total number of arcs in A describing all sentences with length in $range(L(S_i), r)$, and $F(V(f_y), range(L(S), r), None)$ corresponds to the frequency of $V(f_y)$ in the whole set of arcs describing all sentences with length in $range(L(S), r)$.

[1] In all the experiments reported in this work, we set r=2.

Following [12], the *ArcPOSFeat* feature is described as follows:

$$\begin{aligned} &Weight(A_i, ArcPOSFeat, S, C, r) = \\ &\frac{F((Pd, Ph, t))}{F((Pd, X, t))} \cdot \\ &\cdot \frac{F((Pd, Ph, t))}{F((X, Ph, t))} \cdot \\ &\cdot \frac{F(((Pd, Ph, t)(Ph, Ph2, t2)))}{F((Pd, Ph, t))} \cdot \\ &\cdot \frac{F(((Pd, Ph, t)(Ph, Ph2, t2)))}{F((Ph, Ph2, t2))} \cdot \\ &\cdot \frac{F(((Pd, Ph, t)(Ph, Ph2, t2)))}{F((((Pd, X, t))(X, Ph2, t2)))}, \end{aligned}$$

where the triple (Pd, Ph, t) is the arc A_i in which Pd and Ph are the part–of–speech of the dependent d and of the head h respectively, and t is the type of the dependency linking d to h; X is a variable; $F(x)$ is the frequency of x in A; and $((Pd, Ph, t)(Ph, Ph2, t2))$ represents the sequence of two consecutive arcs (going from d to the father of h in A_i) within the dependency tree describing S.

3 The Baselines

For the evaluation of LISCA, different increasingly complex baseline models were selected.

The first baseline consists of a **Random Selection** (henceforth, *RS*) of dependency arcs from the test set. This baseline is calculated in terms of the scores of the parsing systems against the test set.

The second baseline, named **Length Selection** (henceforth, *LS*), is represented by the ordering of arcs by dependency length. This is a strong unsupervised baseline since, as demonstrated in [25], long dependency relations are harder to analyse using statistical dependency parsers than short ones. This is particularly true of transition–based Shift–Reduce parsers (like the one used in the reported experiments) which, still according to [25], appear to be particularly reliable in analysing short dependency relations, due to the specific parsing algorithm that constructs a dependency tree by performing a sequence of parser actions (or transitions) through a greedy parsing strategy. From this, it follows that the ranking of arcs by ascending length potentially represents a highly competitive baseline.

The third and most advanced baseline, hereafter referred to as **dependency TRiplet** (henceforth, *TR*) baseline, is inspired by [29], [30] and [26]. The *TR* baseline is based on parse features, in particular on triplets representing dependency relations as (fd, fh, t), where fd refers to the dependent, fh to the syntactic head and t is the relation type linking d to h.

In this work, we used three different *TR* baselines: *i)* Lexical dependency triplet (henceforth, *LTR*) baseline, where fd and fh are the word–forms of the dependent and of the head; *ii)* Part–of–speech dependency triplet (henceforth, *PTR*), where fd and fh represent the part–of–speech (PoS) of d and h respectively, and *iii)* a combination of LTR and PTR, named *LPTR*, where lexical and PoS information are combined together. Under this baseline, the ranking of dependency arcs is obtained by assigning to each dependency triplet a score calculated using the normalized point–wise mutual information score [6], defined as:

$$NPMI((fd, fh, t)) = \frac{(log(\frac{p((fd,fh,t))}{p((X,fh,t))p((fd))}))}{-log(p((fd, fh, t)))},$$

where $p(a)$ is the probability of the arc a and X is a variable. The probability is computed using frequencies extracted from a large automatically parsed corpus. The *LPTR* baseline score is computed as the product of the *LTR* and *PTR* scores.

4 Experiments

The experiments were organised as follows. First, a wide corpus representative of a new target domain was automatically PoS tagged and dependency–parsed. After this pre–processing stage, the LISCA and baseline algorithms were used to extract statistics from this corpus to be exploited for assigning a quality score to each arc of a newly parsed sentence belonging to the same domain as the automatically parsed target corpus. On the basis of the computed quality scores, a ranking of dependency arcs by decreasing reliability is obtained. The ranking of arcs generated by LISCA and those produced by the other baselines were compared and evaluated.

In particular, two sets of experiments were devised to test the performance of LISCA on corpora belonging to two different domains, with the final aim of showing effectiveness and robustness of the proposed arc ranking algorithm.

4.1 Experimental Setup

The two sets of performed experiments differ at the level of the used datasets, represented respectively by the chemical (CHEM) and biomedical (BIO) datasets of abstracts originally developed for the Domain Adaptation track of the CoNLL 2007 Shared Task [28]. The unlabelled CHEM data consists of 10,482,247 tokens (396,128 sentences) and the gold test set is made of 5,001 tokens (195 sentences); the unlabelled BIO data consists of 9,776,890 tokens (375,421 sentences) and the gold test set is made of 5,017 tokens (200 sentences).

As parsing training data we used the CoNLL 2007 dependency–based version of Sections 2–11 of the Wall Street Journal (WSJ) partition of the Penn Treebank (PTB) [23], for a total of 447,000 tokens and about 18,600 sentences. For testing the parser performances, we used the test set distributed in the Multilingual Track of the ConLL 2007 Shared Task, i.e. the subset of Section 23 of WSJ consisting of 5,003 tokens (214 sentences).

For the linguistic pre–processing of the CHEM and BIO corpora, we used the PoS–tagger described in [11], and the dependency parser DeSR using Multi–Layer Perceptron (MLP) as learning algorithm [5] which represents a state–of–the–art linear–time Shift–Reduce dependency parser (following a “stepwise” approach, [7]). Note that in both cases we relied on the CoNLL 2007 corpora tokenization.

4.2 Evaluation Methodology

The performance of the LISCA algorithm was evaluated with respect to the accuracy of ranked arcs, expressed in terms of the “Labelled Attachment Score” (LAS) obtained by the parser, i.e. the percentage of tokens for which it predicted correct head and dependency relation.

In particular, two types of evaluation were devised both operating on the list of arcs ordered by decreasing reliability generated by the different dependency ranking algorithms. Within the first evaluation type the LAS score was computed for 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.). In the second type of evaluation the ranked list of 5,000–tokens of the test set has been partitioned into sets of 500–tokens each, resulting into 10 groups: for each group, the LAS score has been computed.

In both cases, the evaluation of the ranking algorithms was carried out *1)* by taking into account all dependency relations generated by the parser for the test set, and *2)* by excluding the *ROOT* arc. This double evaluation was devised to make achieved results comparable with the *LS* baseline[2]. This also explains why the plots reported in subfigures 1(a), 1(b), 2(a) and 2(b), representing the results achieved with the first configuration, do not include the *LS* baseline.

5 Results and Discussion

Table 1 reports the accuracy of DeSR against the PTB, BIO and CHEM test sets. By comparing the results obtained on PTB with respect to CHEM and BIO, it can be noted that the parser accuracy decreases significantly. For both CHEM and BIO domains, DeSR has a drop of about 7.5% of LAS and of about 6% and 7% of UAS respectively. This is due to the fact that the domains of CHEM and BIO test sets differ significantly from the training set of the parser. As already pointed out in Section 3, the CHEM and BIO results reported here represent the Random Selection (RS) baseline.

Table 1. DeSR results on PTB, CHEM and BIO

Test corpus	LAS	UAS
PTB	86.09%	87.29%
CHEM	78.50%	81.10%
BIO	78.65%	79.97%

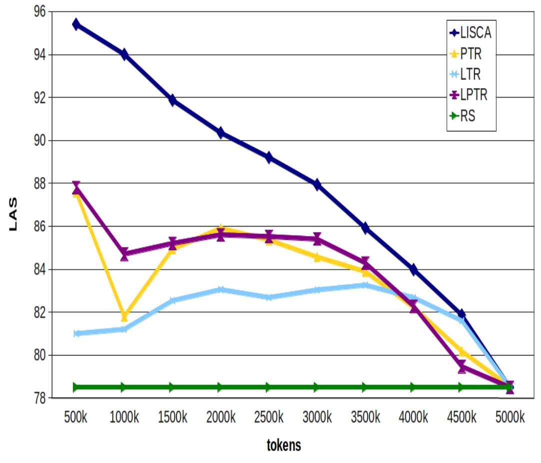
Henceforth, the results obtained in both types of evaluation will be reported in terms of LAS only, since this represents the standard evaluation metric for dependency parsing.

[2]This follows from the fact that dependency length cannot be computed for the artificial *ROOT* node.

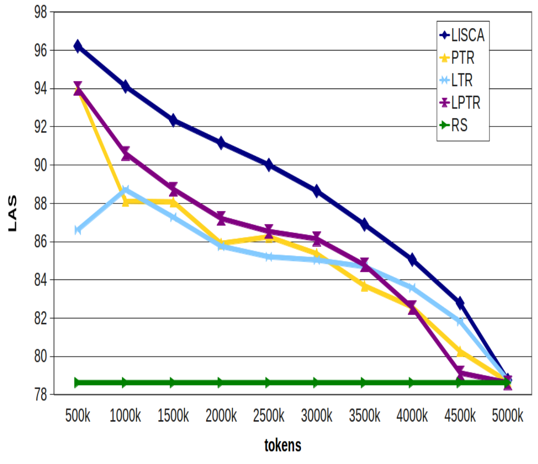
Figure 1 reports the results of the first type of evaluation for all ranking algorithms (LISCA and the baselines), i.e. with respect to the increasingly wider top lists of *k* tokens. Plots in the top row report the results achieved with the first configuration, i.e. by considering all dependency types including the *ROOT* arc. It is interesting to note that all ranking algorithms perform better than the Random Selection (RS) baseline, i.e. all top lists (for each *k* value) show a LAS which is higher than the accuracy of DeSR against the test sets. This demonstrates that, besides RS, all other considered baselines are competive. As the plots in Figure 1 show, LISCA outperforms all baselines for all the *k* token top lists in both configurations (i.e. with and without *ROOT*). For instance, in the first top list (with $k = 500$) LISCA shows an improvement with respect to the best baseline of 7.6% and 6.6% (with and without the *ROOT* respectively) for CHEM, and of 2.2% and 3% for BIO. Besides the improvement observed with respect to the baselines, it is also worth noting that in the first 500–token top lists for both CHEM and BIO, LISCA shows a LAS higher than 95% in all reported experiments, and that in the first 2,000–token top lists the LAS is lower but still high, ranging between 90 and 91%, with an increment of 12–13% with respect to the accuracy of the parser on the CHEM and BIO test sets (RS baseline).

Among all considered baselines, *LS* turned out to be a strong baseline, outperforming all the other baselines for CHEM (see subfigure 1(c)). This could also be seen as following from a characterizing feature of the DeSR parsing algorithm, i.e. its high reliability in analysing short distance dependency relations. Concerning the *TR* baselines, in all performed experiments *LPTR* appears to perform better than the two other baselines, i.e. *PTR* and *LTR*: among them, *LTR* turned out to be always the worst performing one. This comes out clearly in the first top list (with $k = 500$): such a result might be due to the specific behaviour of the point–wise mutual information association score and to data sparseness in the automatically parsed corpus (it is a widely acknowledged fact that mutual information scores are unreliable for low frequency events).

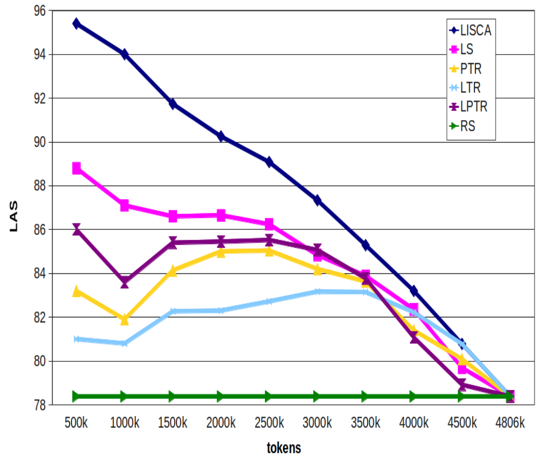
Consider now the second evaluation type: Figure 2 reports the LAS results for all ranking algorithms with respect to each of the ten 500–tokens groups. As in the previous case, histograms in the top row report the results achieved by also considering the



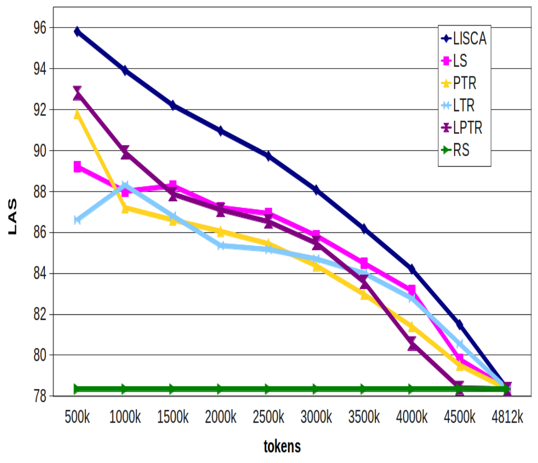
(a) LAS for CHEM with *ROOT*.



(b) LAS for BIO with *ROOT*.



(c) LAS for CHEM without *ROOT*.

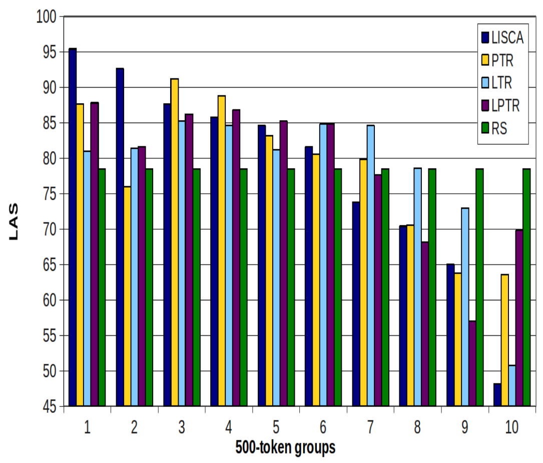


(d) LAS for BIO without *ROOT*.

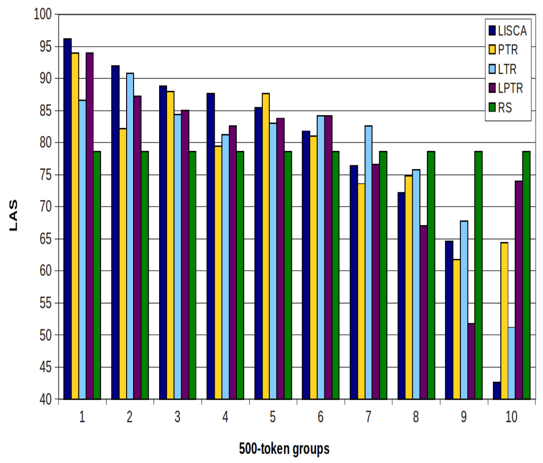
Fig. 1. First evaluation type: LAS of ranking algorithms achieved with the two different configurations

ROOT arc. Similarly to what observed previously, Figure 2 shows that all ranking algorithms perform better than the Random Selection (RS) baseline: all bars in the first half of each histogram are higher than the RS bar, while in the second half the situation is reversed, i.e. the RS bar is taller than the bars representing all other ranking algorithms. Remarkably, the LAS of LISCA ranges from 95% to 96% in the first 500–token group, and from 92% to 92.6% in the second group for both CHEM and BIO domains, thus showing an improvement of 13–17% with respect to the RS baseline. On the other hand, it can be observed that in the last 500–token group RS has a LAS higher than LISCA of 30–45%, thus demonstrating that LISCA is able to discriminate between correct and incorrect arcs.

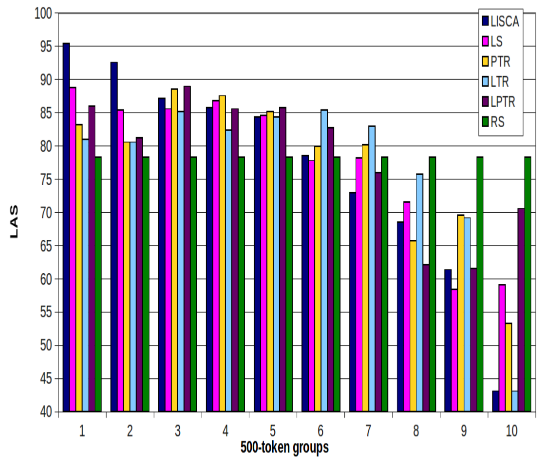
The monotonic decreasing trend of LISCA shows its effectiveness in ranking dependency arcs according to their reliability. As it can be noted in Figure 2, differently from the other baseline algorithms, in the first half of the histogram each bar representing LISCA is shorter than the previous one. It is also interesting to note that LISCA outperforms the LS baseline even in the first and last 500–token groups, where LS gathers respectively very short and very long links which, according to [25], represent excellent indicators of the accuracy of the analysis. This is shown in Table



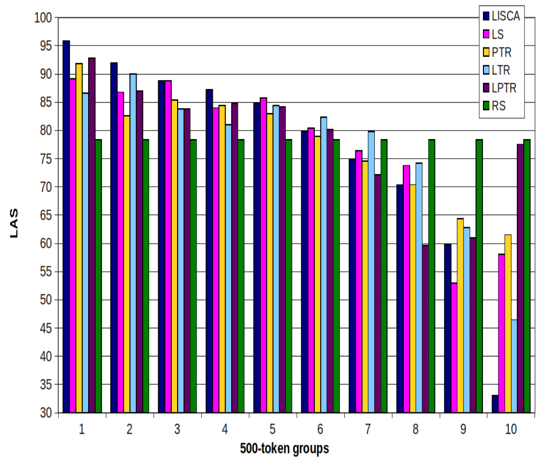
(a) LAS for CHEM with ROOT.



(b) LAS for BIO with ROOT.



(c) LAS for CHEM without ROOT.



(d) LAS for BIO without ROOT.

Fig. 2. Second evaluation type: LAS of ranking algorithms achieved with the two different configurations

2, where the average length of all dependency links (excluding the *ROOT* arc) for all baselines for each 500–token group is reported. For example, the average length of the arcs ordered by LS baseline is equal to 1 for the first four groups of both CHEM and BIO, whereas in the last groups it is equal to 15.21 and 14.3 for CHEM and BIO respectively. Interestingly, the average length of the arcs ordered by LISCA for the CHEM and BIO top lists is close to the average length of the whole CHEM and BIO test sets (i.e. the average length reported in the *RS* column): this fact can be seen as a proof of the LISCA ability to capture domain–specific peculiarities.

6 Conclusion

In this paper we presented LISCA, an unsupervised linguistically–driven algorithm aimed at assigning a quality score to each arc generated by a dependency parser in order to produce a decreasing ranking of arcs from correct to incorrect ones. LISCA exploits statistics about a set of linguistically–motivated and dependency–based features extracted from a large corpus of automatically parsed sentences and uses them to assign a quality score to each arc of a newly parsed sentence belonging to the same domain of the automatically parsed corpus. This

Table 2. Average length of all arcs (excluding the *ROOT*) for all ranking algorithms within each 500–token group

Group	LISCA	LS	PTR	LTR	LPTR	RS
CHEM						
1	2.61	1	7.22	3.38	5.64	3.2
2	2.55	1	4.01	2.82	4.31	3.2
3	2.28	1	1.92	3.69	2.52	3.2
4	2.44	1	2.05	3.22	2.70	3.2
5	2.05	1.11	2.88	3.26	2.77	3.2
6	2.38	2	2.68	2.93	2.65	3.2
7	2.69	2.21	3.31	2.72	3.05	3.2
8	3.22	3.29	3.57	3.03	3.45	3.2
9	3.97	5.88	2.45	2.77	3.23	3.2
10	7.95	15.91	2.73	4.16	2.04	3.2
BIO						
1	3.26	1	7.25	2.89	5.48	3.13
2	2.13	1	3.26	3.22	4.23	3.13
3	2.57	1	2.42	3.65	2.66	3.13
4	2.08	1	2.87	3.18	2.60	3.13
5	2.38	1.19	2.51	3.35	3.06	3.13
6	1.99	2	3.49	2.81	2.89	3.13
7	2.72	2.18	2.94	2.79	2.86	3.13
8	2.91	3.24	2.22	2.86	2.76	3.13
9	4.16	5.44	2.64	3.06	3.17	3.13
10	7.06	14.30	2.53	3.48	2.04	3.13

approach is thus independent from the parsing algorithm used.

LISCA has been tested on two datasets belonging to domains differing from the parser training set. The decreasing ranking produced by LISCA for each dataset was compared and evaluated against the ranking produced by a number of competitive baselines. LISCA turned out to outperform all the considered baselines including *i)* the ordering of arcs by increasing dependency length (*LS*) which represents a strong unsupervised baseline in a dependency parsing scenario due to the fact that long dependency relations are harder to parse than short ones, and *ii)* the dependency TRiplet (TR) baselines successfully used in previous studies to select reliable arcs to improve the parser accuracy.

One of the main qualifying features of LISCA consists in its unsupervised nature: i.e. LISCA does not need manually–annotated training data whose construction is highly expensive and time–consuming. Within unsupervised approaches to the ranking of dependency arcs by reliability, we have demonstrated that LISCA represents an improvement with respect to approaches proposed so far in the literature.

Table 3. Results of the pilot experiment of incorrect arc detection

Arcs (% of total arcs)	Incorrect arcs detected (% of total incorrect arcs)	
	BIO	CHEM
500 (10%)	289 (26.98%)	259 (24.09%)
1000 (20%)	472 (49.29%)	433 (40.28%)
1500 (30%)	608 (56.77%)	582 (54.14%)

Although this is beyond the scope of this paper, it is worth pointing out here that a number of different applications could in principle benefit from the ranking of arcs produced by LISCA. For instance, ranking algorithms are used to improve the accuracy of a dependency parser in a self–training scenario as proposed by [30], [29], [9] and [26] or in an automatic error detection and correction scenario as demonstrated by [18], [3] and [2]. Moreover, the unsupervised nature of LISCA relying on linguistic features extracted from automatically parsed data could be usefully exploited to capture domain–specific peculiarities of the correct arcs generated by the parser as well as to identify problematic linguistic areas for parsers with respect to a given domain. This property of LISCA makes it also a reliable support for human annotators in the construction of domain–specific treebanks, minimizing their annotation efforts. Similarly, LISCA could be used in the construction of treebanks for less–resourced languages.

In order to empirically support this claim, we carried out a pilot experiment aimed at showing effectiveness and reliability of LISCA in an application scenario: in particular, we used LISCA for detecting incorrect arcs within the output of a parser. The dependency arcs generated by the parser for the CHEM and BIO test sets were first ranked by increasing LISCA scores, i.e. from the lower to the higher values. Such ordering of arcs inversely reflects their reliability. The experiment focused on the first 500, 1000 and 1500 arcs representing 10%, 20% and 30% of each test set respectively. In Table 3, for each group of arcs the number of incorrect arcs detected by LISCA is reported, together with the corresponding percentage with respect to the total number of parser errors in the whole test set. The effectiveness of LISCA is shown by the results obtained. For instance, in the first 1000 arcs (i.e. in the 20% of the test set) the algorithm is able to detect the 49.29% of the total incorrect arcs occurring in BIO and 40.28% in CHEM. It should be noted that these results are respectively 29.29 and 20.28 percentage points higher than a random detection baseline. Such encouraging results show that LISCA could be usefully exploited in real application scenarios: this is the direction we are currently working on.

References

1. **Ambati, B. R., Gupta, M., Husain, S., & Sharma, D. M.** (**2010**). A high recall error identification tool for hindi treebank validation. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10)*. Valleta, Malta, 682–686.
2. **Anguiano, E. H. & Candito, M.** (**2011**). Parse correction with specialized models for difficult attachment types. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*. Edinburgh, United Kingdom, 1222–1233.
3. **Attardi, G. & Ciaramita, M.** (**2007**). Tree revision learning for dependency parsing. In *Proceedings of the Conference on Human Language Technologies (NAACL-HLT 2007)*. Rochester, NY, 388–395.
4. **Attardi, G., Dell'Orletta, F., Simi, M., Chanev, A., & Ciaramita, M.** (**2007**). Multilingual dependency parsing and domain adaptation using desr. In *Proceedings of the CoNLL Shared Task Session of the EMNLP-CoNLL 2007*. Prague, 1112–1118.
5. **Attardi, G., Dell'Orletta, F., Simi, M., & Turian, J.** (**2009**). Accurate dependency parsing with a stacked multilayer perceptron. In *Proceedings of EVALITA, Evaluation of NLP and Speech Tools for Italian*. Reggio Emilia, Italy.
6. **Bouma, G.** (**2009**). Normalized (pointwise) mutual information in collocation extraction. In *Proceedings of Biennial GSCL Conference 2009, Meaning: Processing Texts Automatically*. Tubingen, Gunter Narr Verlag, 31–40.
7. **Buchholz, S. & Marsi, E.** (**2006**). Conll-x shared task on multilingual dependency parsing. In *Proceedings of CoNLL*.
8. **Charniak, E. & Johnson, M.** (**2005**). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL'05)*. Ann Arbor, Michigan, 173–180.
9. **Chen, W., Kazama, J., Uchimoto, K., & Torisawa, K.** (**2009**). Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Singapore, 570–579.
10. **Collins, M.** (**1996**). A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association of Computational Linguistics*. Santa Cruz, CA, 184–191.
11. **Dell'Orletta, F.** (**2009**). Ensemble system for part-of-speech tagging. In *Proceedings of Evalita, Evaluation of NLP and Speech Tools for Italian*. Reggio Emilia, Italy.
12. **Dell'Orletta, F., Venturi, G., & Montemagni, S.** (**2011**). Ulisse: an unsupervised algorithm for detecting reliable dependency parses. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL '11), Association for Computational Linguistics*. Portland, Oregon, 115–124.
13. **Dickinson, M.** (**2010**). Detecting errors in automatically-parsed dependency relations. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*. Uppsala, Sweden, 729–738.
14. **Dickinson, M. & Meurers, W. D.** (**2003**). Detecting inconsistencies in treebank. In *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT)*.
15. **Dickinson, M. & Smith, A.** (**2011**). Detecting dependency parse errors with minimal resources. In *Proceedings of the 12th International Conference on Parsing Technologies (IWPT 2011)*. Dublin, Ireland, 241–252.
16. **Frazier, L.** (**1985**). Syntactic complexity. In **Dowty, D., Karttunen, L., & Zwicky, A.**, editors, *Natural Language Parsing: Psychological, Computational, and Thepretical Perspectives*. Cambridge, Cambridge University Press, 129–189.
17. **Gibson, E.** (**1998**). Linguistic complexity: Locality of syntactic dependencies. *Cognition*, 68(1), 1–76.
18. **Hall, K. & Novák, V.** (**2005**). Corrective modeling for non-projective dependency parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*. Vancouver, British Columbia, Canada, 42–52.
19. **Hawkins, J. A.** (**1994**). *A Performance Theory of Order and Constituency*. Cambridge University Press, Cambridge.

20. **Kawahara, D. & Uchimoto, K.** (**2008**). Learning reliability of parses for domain adaptation of dependency parsing. In *Proceedings of IJCNLP 2008*. 709–714.

21. **Lin, D.** (**1996**). On the structural complexity of natural language sentences. In *Proceedings of COLING 1996*. 729–733.

22. **Liu, H.** (**2010**). Dependency direction as a means of word–order typology a method based on dependency treebanks. *Lingua*, 120(6), 1567–1578.

23. **Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B.** (**1993**). Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19(2), 313–330.

24. **McClosky, D., Charniak, E., & Johnson, M.** (**2006**). Reranking and self-training for parser adaptation. In *Proceedings of 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Sydney, Australia, 337–344.

25. **McDonald, R. & Nivre, J.** (**2007**). Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the EMNLP-CoNLL*. 122–131.

26. **Mirroshandel, S. A., Nasr, A., & Roux, J. L.** (**2012**). Semi-supervised dependency parsing using lexical affinities. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Jeju Island, Korea, 777–785.

27. **Ninio, A.** (**1998**). Acquiring a dependency grammar: The first three stages in the acquisition of multiword combinations in hebrew–speaking children. In **Makiello-Jarza, G., Kaiser, J., & Smolczynska, M.**, editors, *Language acquisition and developmental psycology*. Crakow, Universitas.

28. **Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., & Yuret, D.** (**2007**). The conll 2007 shared task on dependency parsing. In *Proceedings of the EMNLP-CoNLL*. 915–932.

29. **Noord, G. V.** (**2007**). Using self-trained bilexical preferences to improve disambiguation accuracy. In *Proceedings of the Tenth International Conference on Parsing Technologies*. Prague, Czech Republic, 1–10.

30. **Plank, B. & Søgaard, A.** (**2012**). Experiments in newswire-to-law adaptation of graph-based dependency parsers. In *Working Notes of EVALITA 2011*. Rome, Italy.

31. **Reichart, R. & Rappoport, A.** (**2009**). Automatic selection of high quality parses created by a fully unsupervised parser. In *Proceedings of CoNLL 2009*. 156–164.

32. **Reichart, R. & Rappoport, A.** (**2009b**). Sample selection for statistical parsers: Cognitively driven algorithms and evaluation measures. In *Proceedings of CoNLL 2009*. 3–11.

33. **Tesniere, L.** (**1959**). *Elements de la syntaxe structurale*. Klincksieck, Paris.

34. **Yngve, V. H.** (**1960**). A model and a hypothesis for language structure. In *Proceedings of the American Philosophical Society*. 444–466.

Felice Dell'Orletta received his Ph.D. degree in Computer Science from the Department of Computer Science of the University of Pisa in 2008. He is currently researcher at the Institute of Computational Linguistics ”Antonio Zampolli” (ILC) of the National Research Council (CNR) in Pisa and member of the ItaliaNLP Laboratory (www.italianlp.it). He has published approx. 50 papers in conferences, workshops and journals. His research interests are mainly focused on probabilistic models of language and statistical natural language processing including part-of-speech tagging, probabilistic dependency parsing, information extraction, named entity recognition and, more in general, machine-learning algorithms for Natural Language Processing and advanced techniques for automatic multilingual text analysis.

Giulia Venturi received her Ph.D. degree in Computational Linguistics from the University of Torino in 2011 with a dissertation entitled “Language and Law: a Computational Linguistics Perspective”. She currently has a Postdoctoral Fellowship at the Institute of Computational Linguistics “Antonio Zampolli” (ILC) of the National Research Council (CNR) in Pisa and she is member of the ItaliaNLP Laboratory (www.italianlp.it). She has published approx. 30 papers in conferences, workshops, journals and books. Her main research interests include extraction of linguistic and extra-linguistic knowledge from domain-specific corpora and its formalization in Domain Ontologies, NLP-based analysis of the legal language with a specific view to the adaptation of NLP tools to the legal language peculiarities, and NLP-based approaches to semantic processing of legal texts.

Simonetta Montemagni is Director of Research at the Institute of Computational Linguistics “Antonio Zampolli” (ILC) of the National Research Council (CNR) in Pisa (Italy), head of the CNR division “Natural Language Processing and Knowledge Management” and member of the ItaliaNLP Laboratory (www.italianlp.it). She holds a Ph.D. in Computational Linguistics from the University of Manchester, Institute of Science and Technology (UK). She worked in the general areas of linguistics and natural language processing since 1986. Her research interests include statistical and robust parsing techniques, acquisition of lexico-semantic information as well as domain knowledge from textual corpora, models of language variation, text mining and ontology learning. She has published over 140 articles, co-edited several books and co-authored two books. She is reviewer for various national and international conferences as well as journals.

Article received on 07/12/2012; accepted on 15/01/2013.