Application of Different Techniques to Dependency Parsing of Basque

Kepa Bengoetxea
IXA NLP Group
University of the Basque Country
Technical School of Engineering, Bilbao,
Plaza La Casilla 3, 48012, Bilbao
kepa.bengoetxea@ehu.es

Koldo Gojenola
IXA NLP Group
University of the Basque Country
Technical School of Engineering, Bilbao,
Plaza La Casilla 3, 48012, Bilbao
koldo.gojenola@ehu.es

Abstract

We present a set of experiments on dependency parsing of the Basque Dependency Treebank (BDT). The present work has examined several directions that try to explore the rich set of morphosyntactic features in the BDT: i) experimenting the impact of morphological features, ii) application of dependency tree transformations, iii) application of a two-stage parsing scheme (stacking), and iv) combinations of the individual experiments. All the tests were conducted using MaltParser (Nivre et al., 2007a), a freely available and state of the art dependency parser generator.

1 Introduction

This paper presents several experiments performed on dependency parsing of the Basque Dependency Treebank (BDT, Aduriz et al., 2003). Basque can be briefly described as a morphologically rich language with free constituent order of the main sentence elements with respect to the main verb.

This work has been developed in the context of dependency parsing exemplified by the CoNLL Shared Task on Dependency Parsing in years 2006 and 2007 (Nivre et al., 2007b), where several systems competed analyzing data from a typologically varied range of 19 languages. The treebanks for all languages were standardized using a previously agreed CoNLL-X format (see Figure 1). An early version of the BDT (BDT I) was one of the evaluated treebanks, which will allow a comparison with our results. One of the conclusions of the CoNLL 2007 workshop (Nivre et al., 2007a) was that there is a class of languages, those that combine a relatively free word order with a high degree of inflection, that obtained the worst scores. This asks for the development of new methods and algorithms that will help to reach the parsing performance of the more studied languages, as English.

In this work, we will take the opportunity of having a new fresh version of the BDT, (BDT II henceforth), which is the result of an extension (three times bigger than the original one), and its redesign (see section 3.2). Using MaltParser, a freely available and state of the art dependency parser for all the experiments (Nivre et al., 2007a), this paper will concentrate on the application of different techniques to the task of parsing this new treebank, with the objective of giving a snapshot that can show the expected gains of each technique, together with some of their combinations. Some of the techniques have already been evaluated with other languages/treebanks or BDT I, while others have been adapted or extended to deal with specific aspects of the Basque language or the Basque Treebank. We will test the following:

- Impact of rich morphology. Although many systems performed feature engineering on the BDT at CoNLL 2007, providing a strong baseline, we will take a step further to improve parsing accuracy taking into account the effect of specific morphosyntactic features.
- Application of dependency-tree transformations. Nilsson et al. (2007) showed that they can increase parsing accuracy across languages/treebanks. We have performed similar experiments adapted to the specific properties of Basque and the BDT.
- Several works have tested the effect of using a two-stage parser (Nivre and McDonald, 2008; Martins et al., 2008), where the second parser takes advantage of features obtained by the first one. Similarly, we will experiment the

Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages, pages 31–39,
Los Angeles, California, June 2010. ©2010 Association for Computational Linguistics

Index	**Word**	**Lemma**	**Category**	**Subcategory**	**Features**	**Head**	**Dependency**
1	etorri	etorri	V	V	_	3	coord
2	dela	izan	AUXV	AUXV	REL:CMP|SUBJ:3S	1	auxmod
3	eta	eta	CONJ	CONJ	_	6	ccomp_obj
4	joan	joan	V	V	_	3	coord
5	dela	izan	AUXV	AUXV	REL:CMP|SUBJ:3S	4	auxmod
6	esan	esan	V	V	_	0	ROOT
7	zien	*edun	AUXV	AUXV	SUBJ:3S|OBJ:3P	6	auxmod
8	mutilak	mutil	NOUN	NOUN_C	CASE:ERG|NUM:S	6	ncsubj
9	.	.	PUNT	PUNT_PUNT	_	8	PUNC

Figure 1: Example of a BDT sentence in the CoNLL-X format

(V = main verb, AUXV = auxiliary verb, CONJ = conjunction, REL = subordinated clause, CMP = completive, ccomp_obj = clausal complement object, ERG = ergative, SUBJ:3S: subject in 3rd person sing., OBJ:3P: object in 3rd person pl, coord = coordination, auxmod = auxiliary, ncsubj = non-clausal subject, ncmod = non-clausal modifier).

addition of new features to the input of the second-stage parser, in the form of morphosyntactic features propagated through the first parser's dependency tree and also as the addition of contextual features (such as category or dependency relation of parent, grandparent, and descendants).

- Combinations of the individual experiments.

The rest of the paper is organized as follows. After presenting related work in section 2, section 3 describes the main resources used in this work. Next, section 4 will examine the details of the different experiments to be performed, while section 5 will evaluate their results. Finally, the last section outlines our main conclusions.

2 Related work

Until recently, many works on treebank parsing have been mostly dedicated to languages with poor morphology, as exemplified by the Penn English Treebank. As the availability of treebanks for typologically different languages has increased, there has been a growing interest towards research on extending the by now standard algorithms and methods to the new languages and treebanks (Tsarfaty et al., 2009). For example, Collins et al. (1999) adapted Collins' parser to Czech, a highly-inflected language. Cowan and Collins (2005) apply the same parser to Spanish, concluding that the inclusion of morphological information improves the analyzer. Eryiğit et al. (2008) experiment the use of several types of morphosyntactic information in Turkish, showing how the richest the information improves precision. They also show that using morphemes as the unit of analysis (instead of words) gets better results, as a result of the agglutinative nature of Turkish, where each wordform contains several morphemes that can be individually relevant for parsing. Goldberg and Tsarfaty (2008) concluded that an integrated model of morphological disambiguation and syntactic parsing in Hebrew Treebank parsing improves the results of a pipelined approach. This is in accord with our experiment of dividing words into morphemes and transforming the tree accordingly (see section 4.2).

Since the early times of treebank-based parsing systems, a lot of effort has been devoted to aspects of preprocessing trees in order to improve the results (Collins, 1999). When applied to dependency parsing, several works (Nilsson et al., 2007; Bengoetxea and Gojenola, 2009a) have concentrated on modifying the structure of the dependency tree, changing its original shape. For example, Nilsson et al. (2007) present the application of pseudoprojective, verbal group and coordination transformations to several languages/treebanks using MaltParser, showing that they improve the results.

Another interesting research direction has examined the application of a two-stage parser, where the second parser tries to improve upon the result of a first parser. For example, Nivre and McDonald (2008) present the combination of two state of the art dependency parsers feeding each another, showing that there is a significant improvement over the simple parsers. This experiment can be seen as an instance of *stacked learning,* which was also tested on dependency parsing of several languages in (Martins et al., 2008) with significant improvements over the base parser.

3 Resources

This section will describe the main resources that have been used in the experiments. First, subsection 3.1 will describe the Basque Dependency Treebank, which has increased its size from 55,469 tokens in its original version to more than 150,000, while subsection 3.2 will present the main characteristics of MaltParser, a state of the art and data-driven dependency parser.

3.1 The Basque Dependency Treebank

Basque can be described as an agglutinative language that presents a high power to generate inflected word-forms, with free constituent order of sentence elements with respect to the main verb. The BDT can be considered a pure dependency treebank from its original design, due mainly to the syntactic characteristics of Basque.

```
(1) Etorri  dela  eta joan  dela   esan  zien mutilak
    come  that-has and go  that-has tell did  boy-the
    The boy told them that he has come and gone
```

Figure 1 contains an example of a sentence (1), annotated in the CoNLL-X format. The text is organized in eight tab-separated columns: word-number, form, lemma, category, subcategory, morphological features, and the dependency relation (headword + dependency). The information in Figure 1 has been simplified due to space reasons, as typically the *Features* column will contain many *morphosyntactic*[1] features (case, number, type of subordinated sentence, …), which are relevant for parsing. The first version of the Basque Dependency Treebank contained 55,469 tokens forming 3,700 sentences (Aduriz et al., 2003). This treebank was used as one of the evaluated treebanks in the CoNLL 2007 Shared Task on Dependency Parsing (Nivre et al., 2007b). Our work will make use of the second version of the BDT (BDT II), which is the consequence of a process of extension and redesign of the original requirements:

- The new version contains 150,000 tokens (11,225 sentences), a three-fold increase.
- The new design considered that all the dependency arcs would connect sentence tokens. In contrast, the original annotation contained empty nodes, especially when dealing with ellipsis and some kinds of coordination. As a result, the number of non-projective arcs diminished from 2.9% in the original treebank to 1.3% in the new version.
- The annotation follows a stand-off markup approach, inspired on TEI-P4 (Artola et al., 2005). There was a conversion process from a set of interconnected XML files to the CoNLL-X format of the present experiments.

Although the different characteristics and size of the two treebank versions do not allow a strict comparison, our preliminary experiments showed that the results on both treebanks were similar regarding our main evaluation criterion (Labeled Attachment Score, or LAS). In the rest of the paper we will only use the new BDT II.

3.2 MaltParser

MaltParser (Nivre et al. 2007a) is a state of the art dependency parser that has been successfully applied to typologically different languages and treebanks. While several variants of the base parser have been implemented, we will use one of its standard versions (MaltParser version 1.3). The parser obtains deterministically a dependency tree in linear-time in a single pass over the input using two main data structures: a stack of partially analyzed items and the remaining input sequence. To determine the best action at each parsing step, the parser uses history-based feature models and discriminative machine learning. In all the following experiments, we will make use of a SVM classifier. The specification of the configuration used for learning can in principle include any kind of column in Figure 1 (such as word-form, lemma, category, subcategory or morphological features), together with a feature function. This means that a learning model can be described as a series of (*column, function*) pairs, where *column* represents the name of a column in Figure 1, and *function* makes reference to the parser's main data structures. For example, the two pairs (Word, Stack[0]), and (Word, Stack[1]) represent two features that correspond to the word-forms on top and next to top elements of the stack, respectively, while (POSTAG, Input[0]) represents the POS category of the first token in the remaining input sequence.

4 Experiments

The following subsections will present three types of techniques that will be tested with the aim of

[1] We will use the term *morphosyntactic* to name the set of features attached to each word-form, which by the agglutinative nature of Basque correspond to both morphology and syntax.

improving the results of the syntactic analyzer. Subsection 4.1 presents the process of fine-tuning the rich set of available morphosyntactic features. Then, 4.2 will describe the application of three types of tree transformations, while subsection 4.3 will examine the application of propagating syntactic features through a first-stage dependency tree, a process that can also be seen as an application of stacked learning, as tested in (Nivre and McDonald, 2008; Martins et al., 2008)

4.1 Feature engineering

The original CoNLL-X format uses 10 different columns (see Figure 1[2]), grouping the full set of morphosyntactic features in a single column. We will experiment the effect of individual features, following two steps:

- First, we tested the effect of incorporating each individual lexical feature, concluding that there were two features that individually gave significant performance increases. They were syntactic case, which is relevant for marking a word's syntactic function (or, equivalently, the type of dependency relation), and subordination type (REL henceforth). This REL feature appears in verb-ending morphemes that specify a type of subordinated sentence, such as in relative, completive, or indirect interrogative clauses. The feature is, therefore, relevant for establishing the main structure of a sentence, helping to delimit main and subordinated clauses, and it is also crucial for determining the dependency relation between the subordinated sentence and the main verb (head).
- Then, we separated these features in two independent columns, grouping the remaining features under the *Features* column. This way, Maltparser's learning specification can be more fine-grained, in terms of three morphosyntactic feature sets (CASE, REL and the rest, see Table 2).

This will allow us testing learning models with different configurations for each column, instead of treating the full set of features as a whole. So, we will have the possibility of experimenting with richer contexts (that is, advancing the Stack and/or Input[3] functions for each feature).

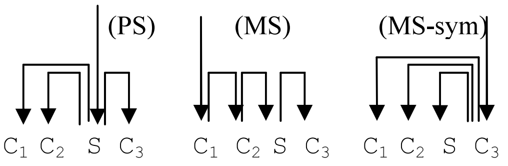

Figure 2. Dependency structures for coordination.

4.2 Tree transformations

Tree transformations have long been applied with the objective of improving parsing results (Collins, 1999; Nilsson et al., 2007). The general process consists of the following steps:

- Apply tree transformations to the treebank
- Train the system on the modified treebank
- Apply the parser to the test set
- Apply the inverse transformations
- Evaluate the result on the original treebank

We will test three different tree transformations, which had already been applied to the Treebank (BDT I) (Bengoetxea and Gojenola, 2009a):

- Projectivization (T_P). This is a language independent transformation already tested in several languages (Nivre and Nilsson, 2005). This transformation is totally language independent, and can be considered a standard transformation. Its performance on the first version of BDT had been already tested (Hall et al., 2007), giving significant improvements This is in accordance with BDT I having a 2.9% of non-projective arcs.
- Coordination (T_C). The transformation on coordinated sentences can be considered general (Nilsson et al., 2007) but it is also language dependent, as it depends on the specific configurations present in each language, mainly the set of coordination conjunctions and the types of elements that can be coordinated, together with their morphosyntactic properties (such as head initial or final). Coordination in BDT (both versions) is annotated in the so called Prague Style (PS, see Figure 2), where the conjunction is taken as the head, and the

[2] As a matter of fact, Figure 1 only shows 8 columns, although the CoNLL-X format includes two additional columns for the projective head (PHEAD) and projective dependency relation (PDEPREL), which have not been used in our work.

[3] Maltparser allows a rich set of functions to be specified for each column. In our experiments we mainly used the Stack and Input functions, which allow the inspection of the contents of the top elements of the Stack (Stack[0], Stack[1], …) or the currently unanalyzed input sequence (Input[0], Input [1], …).

conjuncts depend on it. Nilsson et al. (2007) advocate the Mel´cuk style (MS) for parsing Czech, taking the first conjunct as the head, and creating a chain where each element depends on the preceding one. Basque is a head final language, where many important syntactic features, like case or subordinating morphemes are located at the end of constituents. For that reason, Bengoetxea and Gojenola (2009a) proposed MS-sym, a symmetric variation of MS in which the coordinated elements will be dependents of the last conjunct (which will be the head, see Figure 2).

- Transformation of subordinated sentences (TS). They are formed in Basque by attaching the corresponding morphemes to the auxiliary verbs. However, in BDT (I and II) the verbal elements are organized around the main verb (semantic head) while the syntactic head corresponds to the subordination morpheme, which appears usually attached to the auxiliary. Its main consequence is that the elements bearing the relevant information for parsing are situated far in the tree with respect to their head. In Figure 3, we see that the morpheme *–la*, indicating a subordinated completive sentence, appears down in the tree, and this could affect the correct attachment to the main verb (*esan*). Figure 4 shows the effect of transforming the original tree in Figure 3. The subordination morpheme (*-la*) is separated from the auxiliary verb (*da*), and is "promoted" as the syntactic head of the subordinated sentence. New arcs are created from the main verb (*etorri*) to the morpheme (which is now the head), and also a new dependency relation (SUB).

Overall, the projectivization transformation (T_P) is totally language-independent. T_C (coordination) can be considered in the middle, as it depends on the general characteristics of the language. Finally, the transformation of subordinated sentences (T_S) is specific to the treebank and intrinsecally linked to the agglutinative nature of Basque. Bengoetxea and Gojenola (2009a) also found that the order of transformations can be relevant. Their best system, after applying all the transformations, obtained a 76.80% LAS on BDT I (2.24% improvement over a baseline of 74.52%) on the test set. We include these already evaluated transformations in the present work with two objectives in mind:

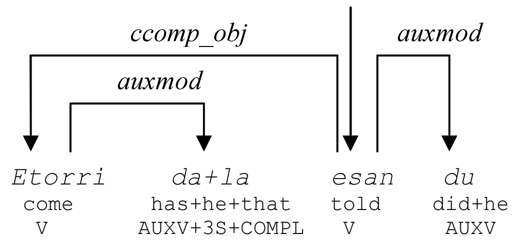

Figure 3. Dependency tree for the sentence *Etorri dela esan du* (He told that he would come).

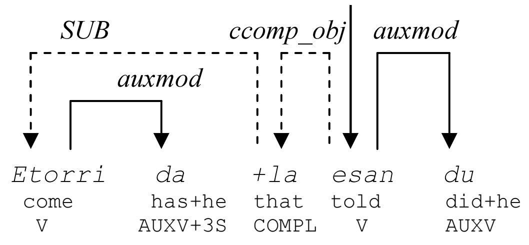

Figure 4. Effect of applying the transformation on subordinated sentences to the tree in Figure 3 (dotted lines represent the modified arcs).

- We want to test its effect on BDT II, 3 times larger than BDT I, and also with a lower proportion of non-projective arcs (1.3%).
- We are also interested in testing its combination with the rest of the techniques (see subsections 4.1 and 4.3).

4.3 Two-stage parsing (stacking)

Bengoetxea and Gojenola (2009b) tested the effect of propagating several morphosyntactic feature values after a first parsing phase, as in classical unification-based grammars, as a means of propagating linguistic information through syntax trees. They applied three types of feature propagation of the morphological feature values: a) from auxiliary verbs to the main verb (verb groups) b) propagation of case and number from post-modifiers to the head noun (noun phrases) c) from the last conjunct to the conjunction (coordination). This was done mainly because Basque is head final, and relevant features are located at the end of constituents.

Nivre and McDonald (2008) present an application of stacked learning to dependency parsing, in which a second predictor is trained to improve the performance of the first. Martins et al. (2008) specify the following steps:

- Split training data D into L partitions D^1, ... , D^L.
- Train L instances of the level 0 parser in the following way: the l-th instance, g^l, is trained

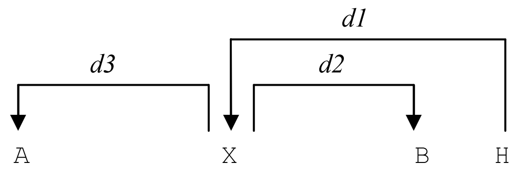

Figure 5. Stacked features. X can take several features from its descendants (dependency arcs d2 and d3) or his head (d1).

on $D^{-l} = D \setminus D^{l}$. Then use g^{l} to output predictions for the (unseen) partition D^{l}. At the end, we have an augmented dataset D^{*} = D + new set of stacked/propagated features.

- Train the level 0 parser g on the original training data D.
- Train the level 1 parser on the augmented training data D^{*}.

In our tests, it was enough with two partitions (L = 2), as experiments with L > 2 did not give any significant improvement. Figure 5 shows the types of information that can be added to each target element. The token X can take several kinds of information from its children (A and B) or his parent (H). The information that is propagated can vary, including part of speech, morphosyntactic features or the dependency relations between X and its children/parent. We can roughly classify the stacked features in two different sets:

- *Linguistic* features (feature passing), such as case or number, which are propagated applying linguistic principles, such as "*the syntactic case is propagated from the dependents towards the head of NPs and postpositional phrases*". The idea is to propagate several *morphosyntactic* features (case, number, …) from dependents to heads.
- *Parser* features. They will be based solely on different dependency tree configurations (see Figure 5), similarly to (Nivre and McDonald, 2008; Martins et al., 2008). Among them, we will test the inclusion of several features (dependency relation, category and morphosyntactic features) from the following: parent, grandparent, siblings, and children.

In the present work, we have devised the following experiments:

- We will test the effect of propagating *linguistic* features on the new BDT II. In contrast to (Bengoetxea and Gojenola, 2009b), who used the enriched *gold* data as D^{*} directly, we will test Martins et al.'s proposal, in which the level 1 parser will be able to learn on the errors of the level 0 parser.
- We will extend these experiments with the use of different *parser* features (Nivre and McDonald, 2008; Martins et al., 2008).

4.4 Combination

Finally, we will combine the different techniques. An important point is to determine whether the techniques are independent (and accumulative) or it could also be that they can serve as alternative treatments to deal with the same phenomena.

5 Evaluation

BDT I was used at the CoNLL 2007 Shared Task, where many systems competed on it (Nivre et al., 2007b). We will use Labeled Attachment Score (LAS) as the evaluation measure: the percentage of correct arcs (both dependency relation and head) over all arcs, with respect to the gold standard. Table 1 shows the best CoNLL 2007 results on BDT I. The best system obtained a score of 76.94%, combining six variants of MaltParser, and competing with 19 systems. Carreras (2007) and Titov and Henderson (2007) obtained the second and third positions, respectively. We consider the last two lines in Table 1 as our baselines, which consist in applying a single MaltParser version (Hall et al., 2007), that obtained the fifth position at CoNLL 2007. Although Hall et al. (2007) applied the projectivization transformation (T_P), we will not use it in our baseline because we want to evaluate the effect of multiple techniques over a base parser. Although we could not use the subset of BDT II corresponding to BDT I, we run[4] a test with a set of sentences the size of BDT I. As could be ex-

		System		LAS
BDT I	**CoNLL 07**	Nivre et al. 2007b (MaltParser, combined)		76.94%
		Carreras, 2007		75.75%
		Titov and Henderson, 2007		75.49%
		Hall et al., 2007 (MaltParser (single parser) + pseudoprojective transformation)		74.99%
	MaltParser (single parser)			74.52%
BDT II	MaltParser (single parser)		BDT I size	74.83%
			Baseline	77.08%

Table 1. Top LAS scores for Basque dependency parsing.

[4] For space reasons, we do not specify details of the algorithm and the parameters. These data can be obtained, together with the BDT II data, from any of the authors.

		Stack[0]	Input[0]	Input[1]	Input[2]
1	**Features**	+	+		
2	**CASE**	+	+	+	
	REL	+	+	+	+
	Features (rest)	+	+		

Table 2. Learning configurations for morphosyntactic features (1 = best model for the whole set of features. 2 = best model when specializing features).

pected, the three-fold increase in the new treebank gives a 2.35% improvement over BDT I.

For evaluation, we divided the treebank in three sets, corresponding to training, development, and test (80%, 10%, and 10%, respectively). All the experiments were done on the development set, leaving the best systems for the final test.

5.1 Single systems

Table 3 shows the results for the basic systems employing each of the techniques advanced in Section 4. As a first result, we see that a new step of reengineering MaltParser's learning configuration was rewarding (see row 2 in Table 3), as morphosyntactic features were more finely specified with respect to the most relevant features. Table 2 presents the baseline and the best learning model[5]. We see that advancing the input lookahead for CASE and REL gives an increase of 0.82 points.

Looking at the transformations (rows 3 to 7), the new Treebank BDT II obtains results similar to those described in (Bengoetxea and Gojenola, 2009a). As could be expected from the reduction of non-projective arcs (from 2.9% to 1.3%), the gains of T_P are proportionally lower than in BDT I. Also, we can observe that T_S alone worsens the baseline, but it gives the best results when combined with the rest (rows 6 and 7). This can be explained because T_S creates new non-projective arcs, so it is effective only if T_P is applied later. The transformation on coordination (T_C) alone does not get better results, but when combined with T_P and T_S gives the best results.

Applying feature propagation and stacking (see rows 9-17), we can see that most of the individual techniques (rows 9-14) give improvements over the baseline. When combining what we defined as *linguistic* features (those morphosyntactic features propagated by the application of three linguistic principles), we can see that their combination seems accumulative (row 15). The *parser* features also give a significant improvement individually (rows 12-14), but, when combined either among themselves (row 16) or with the linguistic features (row 17), their effect does not seem to be additive.

5.2 Combined systems

After getting significant improvements on the individual techniques and some of their combinations, we took a further step to integrate different techniques. An important aspect that must be taken into account is that the combination is not trivial all the times. For example, we have seen (section 5.1) that combinations of the three kinds of tree transformations must be defined having in mind the possible side-effects of any previous transformation. When combining different techniques, care must be taken to avoid any incompatibility. For that reason we only tested some possibilities. Rows 18-21 show some of the combined experiments. Combination of feature optimization with the pseudoprojective transformation yields an accumulative improvement (row 18). However, the combination of all the tree transformations with FO (row 19) does not accumulate. This can be due to the fact that feature optimization already *cancelled* the effect of the transformation on coordination and subordinated sentences, or otherwise it could also need a better exploration of their interleaved effect. Finally, row 21 shows that feature optimization, the pseudoprojective transformation and feature propagation are also accumulative, giving the best results. The relations among the rest of the transformations deserve future examination, as the results do not allow us to extract a precise conclusion.

6 Conclusions and future work

We studied several proposals for improving a baseline system for parsing the Basque Treebank. All the results were evaluated on the new version, BDT II, three times larger than the previous one. We have obtained the following main results:

- Using rich morphological features. We have extended previous works, giving a finer grained description of morphosyntactic features on the learner's configuration,

[5] This experiment was possible due to the fact that MaltParser's functionality was extended, allowing the specification of new columns/features, as the first versions of MaltParser only permitted a single column that included all the features.

		Row	System	LAS	
Single technique	Baseline	1		77.08%	
	Feature optimization	2	FO	*77.90%	(+0.82)
	Transformations	3	T_P	**77.92%	(+0.84)
		4	T_S	75.95%	(-1.13)
		5	T_C	77.05%	(-0.03)
		6	$T_S + T_P$	**78.41%	(+1.33)
		7	$T_S + T_C + T_P$	**78.59%	(+1.51)
	Stacking	9	S_{VG}	**77.68%	(+0.60)
		10	S_{NP}	77.17%	(+0.09)
		11	S_C	77.40%	(+0.32)
		12	S_P	*77.70%	(+0.62)
		13	S_{CH}	*77.80%	(+0.72)
		14	S_{GP}	77.37%	(+0.29)
		15	$S_{VG} + S_{NP} + S_C$	**78.22%	(+1.14)
		16	$S_P + S_{CH}$	**77.96%	(+0.88)
		17	$S_{VG} + S_{NP} + S_C + S_P + S_{CH}$	**78.44%	(+1.36)
Combination		18	FO + T_P	**78.78%	(+1.70)
		19	FO + $T_S + T_C + T_P$	**78.47%	(+1.39)
		20	$T_P + S_{VG} + S_{NP} + S_C$	**78.56%	(+1.48)
		21	FO + $T_P + S_{VG} + S_{NP} + S_C$	**78.98%	(+1.90)

Table 3. Evaluation results.

(FO: feature optimization; T_P T_C T_S: Pseudo-projective, Coordination and Subordinated sentence transformations;
S_{VG}, S_{NP}, S_C: Stacking (feature passing) on Verb Groups, NPs and Coordination;
S_P, S_{CH}, S_{GP}: Stacking (category, features and dependency) on Parent, CHildren and GrandParent;
*: statistically significant in McNemar's test, $p < 0.005$; **: statistically significant, $p < 0.001$)

showing that it can significantly improve the results. In particular, differentiating case and the type of subordinated sentence gives the best LAS increase (+0.82%).

- Tree transformations. We have replicated the set of tree transformations that were tested in the old treebank (Bengoetxea and Gojenola 2009a). Two of the transformations (projectivization and coordination) can be considered language independent, while the treatment of subordination morphemes is related to the morphological nature of Basque.
- Feature propagation. We have experimented the effect of a stacked learning scheme. Some of the stacked features were language-independent, as in (Nivre and McDonald. 2008), but we have also applied a generalization of the stacking mechanism to a morphologically rich language, as some of the stacked features are morphosyntactic features (such as case and number) which were propagated through a first stage dependency tree by the application of linguistic principles (noun phrases, verb groups and coordination).
- Combination of techniques. Although several of the combined approaches are accumulative with respect to the individual systems, some others do not give a improvement over the basic systems. A careful study must be conducted to investigate whether the approaches are exclusive or complementary. For example, the transformation on subordinated sentences and feature propagation on verbal groups seem to be attacking the same problem, i. e., the relations between main and subordinated sentences. In this respect, they can be viewed as alternative approaches to dealing with these phenomena.

The results show that the application of these techniques can give noticeable results, getting an overall improvement of 1.90% (from 77.08% until 78.98%), which can be roughly comparable to the effect of doubling the size of the treebank (see the last two lines of Table 1).

Acknowledgements

This research was supported by the Department of Industry of the Basque Government (IE09-262) and the University of the Basque Country (GIU09/19). Thanks to Joakim Nivre and his team for their support using Maltparser and his fruitful suggestion about the use of stacked features.

References

Itziar Aduriz, Maria Jesus Aranzabe, Jose Maria Arriola, Aitziber Atutxa, Arantza Diaz de Ilarraza, Aitzpea Garmendia and Maite Oronoz. 2003. Construction of a Basque dependency treebank. *Treebanks and Linguistic Theories*.

Xabier Artola, Arantza Díaz de Ilarraza, Nerea Ezeiza, Koldo Gojenola, Gorka Labaka, Aitor Sologaistoa, Aitor Soroa. 2005. A framework for representing and managing linguistic annotations based on typed feature structures. *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, RANLP 2005.

Kepa Bengoetxea and Koldo Gojenola. 2009a. Exploring Treebank Transformations in Dependency Parsing. *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, RANLP'2009.

Kepa Bengoetxea and Koldo Gojenola. 2009b. Application of feature propagation to dependency parsing. *Proceedings of the International Workshop on Parsing Technologies* (IWPT'2009).

Xavier Carreras. 2007. Experiments with a high-order projective dependency parser. In *Proceedings of the CoNLL 2007 Shared Task* (EMNLP-CoNLL).

Shay B. Cohen and Noah A. Smith. 2007. Joint Morphological and Syntactic Disambiguation. *In Proceedings of the CoNLL 2007 Shared Task.*

Michael Collins, Jan Hajic, Lance Ramshaw and Christoph Tillmann. 1999. *A Statistical Parser for Czech*. Proceedings of ACL.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. PhD Dissertation, University of Pennsylvania..

Brooke Cowan and Michael Collins. 2005. Morphology and Reranking for the Statistical Parsing of Spanish. In *Proceedings of EMNLP 2005*.

Gülsen Eryiğit, Joakim Nivre and Kemal Oflazer. 2008. Dependency Parsing of Turkish. *Computational Linguistics*, Vol. 34 (3).

Yoav Goldberg and Reut Tsarfaty. 2008. A Single Generative Model for Joint Morphological Segmentation and Syntactic Parsing. *Proceedings of ACL-HLT* 2008, Colombus, Ohio, USA.

Johan Hall, Jens Nilsson, Joakim Nivre, Gülsen Eryigit, Beáta Megyesi, Mattias Nilsson and Markus Saers. 2007. Single Malt or Blended? A Study in Multilingual Parser Optimization. *Proceedings of the CoNLL Shared Task EMNLP-CoNLL*.

André F. T. Martins, Dipanjan Das, Noah A. Smith, Eric P. Xing. 2008. Stacking Dependency Parsing. *Proceedings of EMNLP-2008*.

Jens Nilsson, Joakim Nivre and Johan Hall. 2007. Generalizing Tree Transformations for Inductive Dependency Parsing. *Proceedings of the 45th Conference of the ACL*.

Joakim Nivre. 2006. *Inductive Dependency Parsing*. Springer.

Joakim Nivre, Johan Hall, Jens Nilsson, Chanev A., Gülsen Eryiğit, Sandra Kübler, Marinov S., and Edwin Marsi. 2007a. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel and Deniz Yuret. 2007b. The CoNLL 2007 Shared Task on Dependency Parsing. *Proceedings of EMNLP-CoNL*L.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. *Proceedings* of ACL-2008.

Ivan Titov and James Henderson. 2007. Fast and robust multilingual dependency parsing with a generative latent variable model. In *Proceedings of the CoNLL 2007 Shared Task* (EMNLP-CoNLL).

Reut Tsarfaty, Khalil Sima'an, and Remko Scha. 2009. An Alternative to Head-Driven Approaches for Parsing a (Relatively) Free Word-Order Language. *Proceedings of EMNLP.*