Design and evaluation of an agreement error detection system: testing the effect of ambiguity, parser and corpus type

Maite Oronoz, Arantza Díaz de Ilarraza, and Koldo Gojenola

IXA NLP Group
University of the Basque Country
{maite.oronoz,a.diazdeilarraza,koldo.gojenola}@ehu.es
http://ixa.si.ehu.es

Abstract. We present a system for the detection of agreement errors in Basque, a language with agglutinative morphology and free order of the main sentence constituents. Due to their complexity, agreement errors are one of the most frequent error types found in written texts. As the constituents concerning agreement can appear in any order in the sentence, we have implemented a system that makes use of dependency trees of the sentence, which abstract over specific constituent orders. We have used *Saroi*, a tool that obtains the analysis trees that fulfill a set of restrictions described by means of declarative rules. This tool is applied to the output of two dependency analyzers: *MaltIxa* (data-driven) and *EDGK* (rule-based). The system has been evaluated on two corpora: a group of texts containing errors, and another one composed of correct texts. As a secondary result, we have also estimated a measure of the impact of syntactic ambiguity on the quality of the results.

Key words: Grammar error, ambiguity, parsing

1 Introduction

Detection of grammatical errors is a relevant area of study in computer-assisted language learning and grammar checking. This paper presents the implementation and evaluation of a system for the detection of agreement errors in Basque, regarded as one of the most frequent kinds of error [1].

Referring to the process of detecting grammatical errors, Díaz de Ilarraza et al. [2] distinguish between *local* and *global* errors depending on the context they appear. Agreement errors belong to the second category due to the fact that the process of finding them is not limited to a local context (that is, a window of five or six consecutive words) but their detection requires the use of full sentence contexts, as the types of elements involving agreement (subject-verb, object-verb, indirect object-verb) may appear far from each other in the sentence. For this reason, our system will make use of syntactic dependency trees, which have the property of abstracting over specific constituent orders.

After the analysis of dependencies the system will make use of *Saroi* [3], a tool that, given a set of dependency trees, obtains those that fulfill the set of restrictions described by means of declarative rules. Although the tool is useful for several types of tree inspection processes, in this work we will use it for the detection of ungrammatical structures. *Saroi* will be applied to the outputs of two dependency analyzers: *EDGK*, a knowledge-based dependency parser [4]; and, *MaltIxa* [5] a data-driven parser based on Maltparser, a freely available and state of the art parser [6]. For the evaluation of the system, texts containing errors and correct texts from the Basque Dependency Treebank [7] will be used.

We are also concerned about the impact of morphosyntactic ambiguity in the quality of our system. A lot of error detection has been carried out on English, for which this kind of ambiguity is less of an issue, but in morphologically rich languages, a deep analysis of the influence of ambiguity in error detection is, in our opinion, fundamental. Among the three main types of ambiguity that can be relevant to grammatical error treatment (morphological, syntactic and semantic), our study will concentrate on measuring the effect of morphological and syntactic ambiguity in the results, leaving aside semantic ambiguity.

The remainder of this paper is organized as follows. After this introduction, section 2 relates our work to similar systems. Section 3 comments on general aspects of agreement errors in Basque. Section 4 will describe the linguistic resources used for the analysis of incorrect texts (corpora), and the main computational tools: two dependency analyzers and *Saroi*, a tool for tree inspection. Section 5 will present the experiments performed and the main results obtained. We conclude the paper in section 6 with our main contributions.

2 A bird's eye view of error detection techniques

Approaches to grammatical error detection/correction are difficult to compare due to mainly the following reasons: i) most of them concentrate on one error type, and ii) the lack of large available error corpora. Choosing the more appropiate technique to the problem of error detection is not a trivial decision. Empirical and knowledge-based approaches can be used for this purpose.

Empirical approaches are suitable for error types related to the omission, replacement or addition of elements. For example, Tetreault and Chodorow [8] use machine learning techniques to detect errors involving prepositions in non-native English speakers. A deeply studied area using machine learning techniques is that of "context-sensitive spelling correction" [9], where the objective is to detect errors due to word confusion (e.g. *to/too*). Bigert and Knutsson [10] prove that precision is significantly improved when unsupervised methods are combined with linguistic information.

Regarding knowledge-based methods, many types of "local syntactic errors" have been detected by means of tools based on finite-state automata or transducers, such as Constraint Grammar (CG) [11], The Xerox Finite State Tool [12] or *ad hoc* systems. Systems based on finite state techniques usually define error patterns encoded in the form of rules which are applied to the analyzed texts.

For "global error" treatment, approaches based on context free grammars (CFG) or finite state techniques have been used. For example, CFG-based systems have experimented with the "relaxation" of some constraints in the grammar [13] or have specially developed error grammars [14]. Statistical parsers have also been used, that get a measure of grammaticality [15].

The relationship between ambiguity and error detection has been mentioned in very few occasions [16, 17]. Similarly to most NLP areas, the development of tools for grammatical error detection finds ambiguity as a main obstacle for the design of efficient and accurate systems. Birn [16] states that the errors accumulated through morphological and syntactic analysis make it difficult to detect grammatical errors.

3 Agreement errors in Basque

Basque is an agglutinative language with free order among the elements of the sentence. When classifying the errors related to agreement, we can distinguish three types of contexts:

– Intra-sentence agreement. The subject, object and indirect object must agree with the verb in case, number and person. These constituents can appear in any order in the sentence. It can appear in simple or compound sentences.
– Intra-phrase agreement. The constituents inside a phrase (e.g., a determiner) must agree with the head of the phrase (e.g., a noun).
– Other types of agreement. For example, an apposition and its corresponding main clause must agree in case, number and person.

**Zentral nuklear-r-ak*	*zakar erradiaktiboa*	*eratzen dute*
Power_station nuclear-0-the/ABS/PL/DET	rubish radioactive/ABS/SG	create AUX/SBJ:ERG_3PL,OBJ:ABS_3SG
'*The nuclear power station'	'create'	'radioactive rubbish'

Table 1. Agreement error (SBJ: subject, OBJ: object, ERG: ergative, ABS: absolutive).

We performed a manual study on the frequency of each type of error over a sample of 64 sentences containing agreement errors (and, sometimes, other types of error) that were taken from a database containing grammatical errors, and we found that intra-sentence agreement was by far the most common type (59 of the sentences, compared to 5 intra-phrase errors). For that reason, we dedicated our effort to this kind of error. Table 1 shows an example of a typical agreement error, where the verb must agree with the main grammatical elements (subject and object) in case, number and person. The fact that these elements can appear in any order with respect to the verb and also to each other makes error detection a difficult task, as there is a high number of possible permutations.

In brief, intra-sentence agreement errors can be abstracted as a local dependency tree where the main verb is the head, and the subject, object and indirect object are the dependents, together with the auxiliary that is also a dependent of the main verb and contains agreement information about the gramatical relations (case, number and definiteness).

4 General linguistic resources

4.1 The Corpus

The task of creating large sets of ungrammatical sentences is a necessary but time consuming activity. A corpus of this type can be composed of sentences produced by language learners (learner corpus) or it can be taken from a general error corpus not necessarily produced in a language-learning context [14]. Although some approaches propose the automatic creation of ungrammatical sentences [18], we decided to use a set of genuine errors. For evaluation, we use two corpora:

– A general purpose error corpus. It contains 1,000,000 words collected from different sources (language schools, technical reports, e-mails. . .). For the current experiments, we took a small subset of this corpus (5,000 words or 267 sentences), in which agreement errors were manually annotated.
– The Basque Dependency Treebank (BDT). This is a collection of presumably correct texts, that contains 55,000 tokens. Working with correct texts allows us to test the system negatively, that is, we test the system's behavior regarding false alarms, an important facet in automatic error detection.

4.2 Syntactic analysis

The creation of NLP tools is a very expensive task, so, instead of preparing specially tailored resources for error processing we decided to use the existing systems in our group, and perform the necessary adaptations to deal with ill-formed sentences. For the analysis of the input texts, we use the syntactic analysis chain for Basque [19]. It is composed of three main components (see figure 1):

– *Morphosyntactic processing.* It includes tokenization, morphological analysis, and detection of multiwords, followed by morphological disambiguation.
– *Chunking.* It detects named entities, and, after a shallow syntactic function disambiguation phase, obtains nominal and verbal chunks.
– *Dependency parsing.* A parser obtains dependency trees.

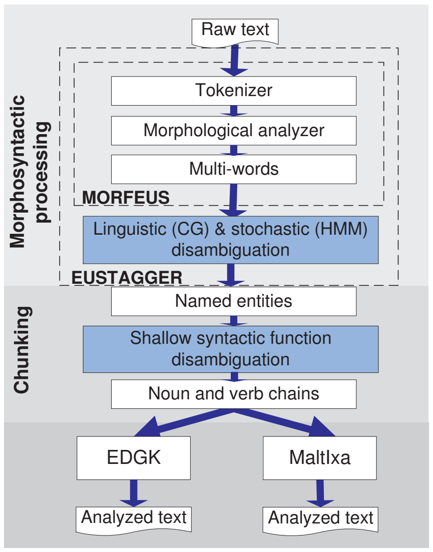
There are two modules in charge of disambiguation (see figure 1):

– *Morphosyntactic disambiguation* (linguistic and stochastic disambiguation). After applying the morphological analyzer (MORFEUS), the tagger/lemmatizer EUSTAGGER obtains the lemma and category of each form, also performing disambiguation using the part of speech (POS), fine grained POS (SubPOS) or case. Disambiguation is carried out by means of linguistic rules using Constraint Grammar (CG) and stochastic techniques [20]. Figure 1 shows the parameterizable disambiguation levels in EUSTAGGER. M1, M2 and M3 combine linguistic and stochastic disambiguation using different linguistic features, while M4 only uses CG. M3 is the option that disambiguates the most (95.42% precision).

– *Shallow syntactic function disambiguation.* This is carried out in two levels, which disambiguate different kinds of functions (see figure 1):
  - S1: it deals with syntactic functions (SF) related to nominal and verbal chunks. That is, functions internal to chunks.
  - S2 treats all functions in S1 plus the main syntactic functions.

Two dependency-based parsers have been used in the present work:

– *EDGK*, a knowledge-based dependency parser [4] based on CG.
– *MaltIxa*, an adaptation of Maltparser, a data-driven dependency parser [6] successfully applied to typologically different languages and treebanks.



Level	Linguistic features	Method
M1	POS	CG + HMM
M2	POS + SubPOS	CG + HMM
M3	POS + SubPOS + Case	CG + HMM
M4	M3 + rest of features	CG
S1	SF nom. & verb. chunks	CG
S2	S1 + main SF	CG

Fig. 1. The syntactic analysis chain for Basque and the disambiguation levels in it.

Regarding accuracy, *EDGK* obtains 48% precision and 46% recall on well-formed texts while *MaltIxa* obtains 76.76% LAS[1]. Although the results are not directly comparable, they serve as an estimate of each parser's performance.

4.3 *Saroi: A tool for inspecting dependency trees*

For the detection of agreement errors we applied *Saroi*, a system developed to apply a set of query-rules to dependency trees. *Saroi* takes as input a group of analysis trees and a group of rules, and obtains as output the dependency trees

[1] *Labeled Attachment Score.*

that fulfill the conditions described in the rules. Its main general objective is the analysis of any linguistic phenomena in corpora.

Figure 2 shows an example of a rule that detects the error in the dependency tree of the same figure. In the sentence the subject *zentral nuklearrak* (nuclear power station), in absolutive case, and the auxiliary verb, *dute* (linked to the main verb *eratzen*, create) and which needs a subject in ergative, do not agree.

```
AGREEMENT_SUBJ_CASE_N_NK (
Detect (
  @!ncsubj!ncmod∼ &
  @!auxmod.type == 'nor-nork' &
  @!ncsubj!ncmod.case !=   )
```

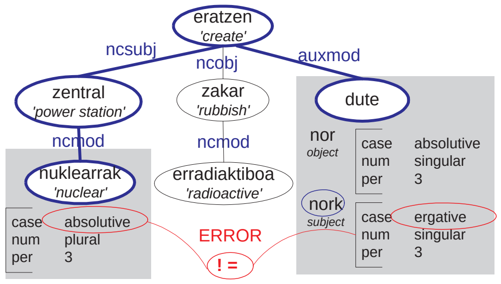


Fig. 2. A rule (left side) detecting the agreement error in the dependency tree (right side) of the sentence in Basque (*Nuclear power station create radioactive rubbish).

Saroi uses as input the result of the syntactic analyzer (see section 4.2), in which the relations between the elements of the sentence are ambiguous, as a result of the remaining morphosyntactic ambiguity (see figure 3 in which, for example, "nuklearrak" has 3 interpretations). Then, *Saroi* constructs all the set of non ambiguous trees starting from an initially ambiguous tree (figure 3). The detection rules are applied to the expanded set of dependency trees.

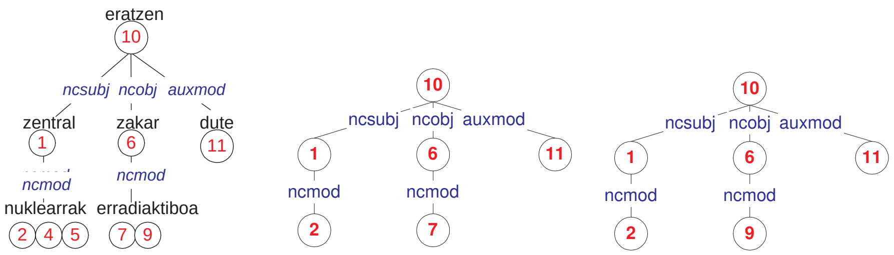


Fig. 3. Ambiguous tree and some of its corresponding non ambiguous trees.

5 Experiments

In this section we will first comment on the experimental settings of the evaluation (5.1 and 5.2), and then we will present the results obtained (5.4).

5.1 Preprocessing

When using the predefined linguistic analysis chain for the detection of agreement errors, we had to take several aspects into account:

– *Difficulties due to language features (ellipsis and ambiguity).* In Basque "The phrases that agree with the verb need not be overtly manifest in the sentence: ergative, dative and absolutive noun phrases or pronouns can be absent and understood[2]". The inherent ambiguity of ellipsis together with semantic ambiguity make it difficult to decide on the correctness of a sentence.
– *Difficulties inherent to automatic language processing.* One of the syntactic analyzers (EDGK) obtains *partial analyses*, that is, not all the elements of the sentence appear in the final dependency trees, due to lack of coverage of the parser. Additionally, the errors mount up in the analysis chain, increasing the number of false alarms.

In an effort to overcome the mentioned problems, we decided to add a preprocessing module that will enrich dependency trees in three ways:

– Enriching nodes corresponding to coordination. For example, when two singular NPs are coordinated, the resulting constituent will agree in plural. For this task we used a set of CG rules.
– Enriching the auxiliary verb with agreement information about case, number and person of the subject, object and indirect object, that was not explicitly shown but was implicitly known. For example, the auxiliary verb *dute* indicates that the subject is *haiek* ('those') and the object *hura* ('that'). We made explicit, for example, that the subject has the features: case= "ergative", number="plural" and person="3".
– Enriching verbs with subcategorization information relevant for agreement, using patterns extracted from three data sources: i) manually developed schemas ii) realization-schemas automatically extracted from a corpus and, iii) information about auxiliary verbs from a dictionary.

5.2 Evaluation methodology

Considering the problems mentioned in section 5.1 and being concerned about the impact of ambiguity in the quality of our analyzers, we followed these steps:

1. We chose the best option for morphological and syntactic disambiguation.
2. Once we decided the appropiate disambiguation level, we evaluated the system using two corpora: correct and error corpora.

An important remark regarding evaluation is that we will not apply the standard development-refinement-test cycle, but instead we will follow a development-test methodology: a) design of error detection rules in Saroi, and, b) evaluation.

[2] http://www.ei.ehu.es/

This means that there will not be a second step for the refinement of the rules after examining their results on a development set. Our aim was to test the effectiveness of a set of *clean* error detecting rules over different settings (corpus, parser and ambiguity). In that respect, the rules examine *clear* (and possibly naive) linguistic statements (e.g. the subject and verb must agree in case and number). This also means that there will be room for improvement of the results, after adapting the error detection rules to the details of real and/or noisy data.

5.3 Election of the disambiguation level

Due to morphosyntactic and syntactic ambiguity, a number of trees ranging from 1 to more than 100 are generated for each sentence. Taking into account the combinations of morphosyntactic and shallow syntactic function disambiguation levels, the best disambiguation criteria should be those that: a) detect the highest number of errors in ungrammatical sentences, b) give the lowest number of false alarms in grammatical sentences, and c) generate the lowest number of analysis trees for each sentence (efficiency). With this objective, we followed two steps:

1. First, we chose the best morphosyntactic disambiguation level.
2. Second, after the morphosyntactic disambiguation level was fixed, we selected the best option for shallow syntactic function disambiguation.

For that reason, we selected a set of 10 ungrammatical sentences and their respective corrections (one for each sentence, that is, a total of 20 sentences). The sentences were analyzed with the eight disambiguation combinations[3] giving the results shown in table 2. The two combinations that generate the lowest number of trees with acceptable detection and false alarm rates were those performing the deepest morphosyntactic disambiguation, that is, M3[4] (S1 and S2).

Disambiguation combinations	M1-S1	M2-S1	**M3-S1**	M4-S1	M1-S2	M2-S2	**M3-S2**	M4-S2
Number of trees	67.7	67.7	**27.8**	46.7	22.11	22.11	**11.6**	11.62
Errors in ungrammatical	5	5	**6**	6	5	5	**6**	6
False alarms in grammatical	0	0	**1**	1	0	0	**1**	0

Table 2. Looking for the best morphosyntactic disambiguation-combination.

Next, we performed a deeper analysis to choose the best syntactic function disambiguation level (S1 or S2). We soon realized that the grammar that assigns the dependency relations to correct texts need of relaxation when applied to ill-formed ones. For example, in the sentence "**nik ez nago konforme*" (I do not agree), the word "*nik*" (I) was not tagged as *subject* as it carries the ergative case, and the auxiliary verb asks for a subject in absolutive (this is a constraint

[3] 8 combinations: 4 morphosyntactic * 2 syntactic.

[4] Although the M4-S2 combination in table 2 seems to be good, it sometimes creates too many trees and, in other cases, it does not obtain any analysis tree.

in the dependency grammar when assigning the *subject* tag). We experimented relaxing all the conditions referred to the type of auxiliary in the rules assigning *subject*, *object* and *indirect object* relations. This relaxation is not performed for error detection (this is done by means of error detection rules) but it is necessary for the assigning of dependency relations to ungrammatical sentences. Then, in a second experiment, we used a set of 75 sentences containing agreement errors together with their corrections. The sentences were analyzed with the M3-S1-Relaxed, M3-S1-NotRelaxed, M3-S2-Relaxed and M3-S2-NotRelaxed combinations. The best results were obtained with the M3-S2-Relaxed option, that is, the option that disambiguates the most and with the relaxed dependency relation assignment. A deeper study about the impact of ambiguity in error detection is described in [2].

5.4 Evaluation of the system

After these tests, we noticed that the results are directly proportional to the parser's accuracy. When the relations are wrongly assigned, the detection of agreement errors is difficult. Sometimes, a false detection occurs, that is, an erroneous sentence is flagged as incorrect, but with a rule that is not the expected one. The rules mark the sentence as incorrect, but they fail in the diagnosis.

Correct corpora. We evaluated our system against the Basque Dependency Treebank. Its relations are presumably perfect (there is no need of a parser, neither the problem of ambiguity nor partial parsing), so the system should perform well. This experiment served to evaluate the system on false alarms. A subset containing 1906 trees was used. After applying the detection rules, 161 errors were flagged (8.45 % of the corpus). As this implies a high false alarm rate, we made a detailed analysis (table 3), finding out that:

– 90 of the cases were due to incorrect tagging, and *could not be considered false alarms*. In 41 of these sentences the error rule was applied because of treebank tagging decisions associated to special phenomena (e.g., in cases of an elliptical verb, two subjects were attached to the same verb when this is not grammatically correct) while in 49, annotation errors were detected (e.g. the object and the subject were mixed up because in Basque sometimes they take the same form. . .). So they correspond to treebank tagging errors.
– In 63 of the cases a *false alarm* (FA) occurs. In the great majority of the cases (58), the FA was flagged due to the lack of information in the verb subcategorization schemas. In these cases the verb appears with an unusual auxiliary verb (with complete subcategorization information, these are likely to disappear). The rest of the false alarms are very specific cases.
– In 8 cases a real agreement error occurs in the treebank.

In short, this experiment, apart from detecting false alarms in the treebank, also served to detect annotation mistakes, and gave us a measure of the importance of having correct verb subcategorization schemas.

Flagged by the system	**Numb.**	**From FA**	**From treebank**
Not considered FA	90	55.9 %	4.72 %
FA	63	39.13 %	**3.30 %**
Real errors	8	4.97 %	
Total	161		8.45 %

Table 3. Evaluation results on the Basque Treebank.

Error corpora. We also performed an evaluation of the system on error corpora, using both *EDGK* and *MaltIxa*. We applied the agreement detection rules to all the possible analysis trees of the sentences. We calculated four results:

1. Using a data-driven parser (*MaltIxa*, **M**).
2. The knowledge-based parser (*EDGK*, **E**).
3. *MaltIxa* **and** *EDGK* (**M & E**). An error will be marked if it is flagged in the dependency trees obtained by *MaltIxa* and *EDGK*.
4. *MaltIxa* **or** *EDGK* (**M | E**). If an error is flagged on the output of either of the syntactic analyzers, the sentence will be deemed erroneous.

Examining the results in table 4 we see that, when applying the full set of error detection rules, precision varies between 24.26% and 26.19%. As could be expected, the best precision results were reached with the option **M & E** (when the error is flagged in the trees analyzed by both analyzers, the system is certain about the error). However, recall falls down (24.44%). In general, looking to both precision and recall, the two best options seem to be **M** and **M | E**. In general, the data-driven parser gets better results with correct texts, and it also behaves better with incorrect sentences, showing a robust behaviour.

There are two error detection rules (named TWO_SUBJ and TWO_OBJ) that account for most of the false alarms, both with EDGK and *MaltIxa*. These rules mark the attachment of two subjects (objects) to a verb. This phenomenon can occur as a consequence of a genuine agreement error, but also because of an incorrect dependency analysis, and is the reason for many false alarms. We think that as the frequency of incorrect analysis trees is relatively high, these rules cause more harm than good. For that reason, we perform three experiments to confirm this assumption. In the second row of table 4 we show the results without considering the rule that detects two subjects (TWO_SUBJ). In the third one, the rule that checks the appearance of two objects is removed (TWO_OBJ) and, finally, the last row shows the result of removing both rules. The best results are obtained in the last case: 44.44% precision in the **M & E** option against the worse recall (17.77%, and f-score of 25.38). Considering precision and recall, *MaltIxa* gives the best results (38.88% precision and 46.66% recall, f-score 42.41).

6 Conclusions and future work

In this work we have presented a set of experiments on agreement error detection applied to an agglutinative and free constituent-order language. For this, we

		Correctly detected	**FA**	**Detected**	**P**	**R**	**F**
All the rules	**M**	28	81	109	**25.68 %**	**62.22 %**	36.35
	E	17	53	70	24.28 %	37.77 %	29.56
	M & E	11	31	42	**26.19 %**	24.44 %	25.28
	M | E	33	103	136	24.26 %	**73.33 %**	36.45
Without the rule	**M**	26	59	85	30.58 %	57.77 %	39.99
TWO_SUBJ	**E**	14	35	49	28.57 %	31.11 %	29.78
	M & E	10	21	31	32.25 %	22.22 %	26.31
	M | E	29	73	102	28.43 %	64.44 %	39.45
Without the rule	**M**	22	55	77	28.57 %	48.88 %	36.06
TWO_OBJ	**E**	12	39	51	23.52 %	26.66 %	24.99
	M & E	8	19	27	29.62 %	17.77 %	22.21
	M | E	26	75	101	25.74 %	57.77 %	35.61
Without the rules	**M**	21	33	54	**38.88 %**	**46.66 %**	42.41
TWO_SUBJ	**E**	10	19	29	34.48 %	22.22 %	27.02
and	**M & E**	8	10	18	**44.44 %**	17.77 %	25.38
TWO_OBJ	**M | E**	23	42	65	**35.38 %**	**51.11 %**	41.81
Number of errors							45
Number of words							4995

Table 4. Agreement error detection with *MaltIxa* and EDGK.

have used *Saroi*, a tool built for the inspection of dependency trees. The tool allows us to design restrictions by means of query-rules to be applied on the output of dependency parsers. In the evaluation we have experimented tuning the ambiguity of the analysis chain, we have used two general-purpose dependency parsers and two types of corpora.

When analyzing the Basque Dependency Treebank, we have detected ill-formed dependency trees, that is, manual annotation mistakes. Additionally, we have evaluated our system regarding false alarms, obtaining a false alarm rate of 3.30 %. Most of the alarms could be easily avoided improving the verb subcategorization schemas we use, and in this way leaving a minimal false alarm rate. In consequence, we think that the precision of our grammar rules is high.

One of the main problems is the lack of coverage of the dependency analyzers. When the trees are not syntactically well-formed, the system is more prone to signal a false alarm. Any improvement in syntactic analysis will have a positive effect on the error detection system. In the future, we want to analyze how complementary are MaltIxa and EDGK, and how they could be combined to obtain suitable analysis trees.

Working with real texts also led us to consider the problem of ambiguity. The best results are obtained when using the deepest disambiguation level (both morphosyntactic and syntactic). This can be explained by the explosion in the number of trees when "all" the ambiguity is considered.

References

1. Ilari Zubiri 1994. *Gramática didáctica del euskara.* Didaktiker, Bilbo.
2. A. Díaz de Ilarraza, K. Gojenola, and M. Oronoz. 2009. Evaluating the Impact of Morphosyntactic Ambiguity in Grammatical Error Detection. In *Recent Advances in Natural Language Processing 2009*, Borovets, Bulgary.

3. A. Díaz de Ilarraza, K. Gojenola, and M. Oronoz. 2005. Design and development of a system for the detection of agreement errors in Basque. In *CICLing2005, Mexico.*
4. M. Aranzabe. 2008. *Dependentzia-ereduan oinarritutako baliabide sintaktikoak: zuhaitz-bankua eta gramatika konputazionala.* Ph.D. thesis, UPV-EHU.
5. K. Bengoetxea and K. Gojenola. 2009. Exploring treebank transformations in dependency parsing. In *RANLP 2009*, Borovets, Bulgaria.
6. J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryiğit, S. Kübler, S. Marinov, and E. Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.
7. I. Aduriz, M. Aranzabe, J. M. Arriola, A. Atutxa, A. Díaz de Ilarraza, A. Garmendia, and M. Oronoz. 2003. Construction of a Basque dependency treebank. In *TLT 2003. Second Workshop on Treebanks and Linguistic Theories*, Vaxjo, Sweden.
8. J.R. Tetreault and M. Chodorow. 2008. The ups and downs of preposition error detection in esl writing. In *Proceedings of Coling,*, Manchester.
9. A. J. Carlson, J. Rosen, and D. Roth. 2001. Scaling up context-sensitive text correction. In *Proceedings of the Thirteenth Innovative Applications of Artificial Intelligence Conference (IAAI-01)*, Menlo Park CA.
10. J. Bigert and O. Knutsson. 2002. Robust error detection: A hybrid approach combining unsupervised error detection and linguistic knowledge. In *Romand 2002*, Italy.
11. F. Karlsson, A. Voutilainen, J. Heikkila, and A. Anttila. 1995. *Constraint Grammar: Language-independent System for Parsing Unrestricted Text.* Berlin.
12. L. Karttunen, T. Gaál, and A. Kempe. 1997. Xerox finite state tool. Technical report, Xerox Research Centre Europe.
13. R. Teixeira Martins, R. Hasegawa, M. Volpe Nunes, G. Montilha, and Jr. Osvaldo Novais De Oliveira. 1998. Linguistic issues in the development of ReGra: A grammar checker for Brazilian Portuguese. *NLE*, 4(4):287–307.
14. J. Foster and C. Vogel. 2004. Good reasons for noting bad grammar: Constructing a corpus of ungrammatical language. In *International Conference on Linguistic Evidence: Empirical, Theoretical and Computational Perspectives*, Tübingen, Germany.
15. J. Wagner and J. Foster. 2009. The effect of correcting grammatical errors on parse probabilities. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, Paris, France.
16. J. Birn. 2000. Detecting grammar errors with Lingsoft's Swedish grammar-checker. In *Proceedings from the 12th Nordiske datalingvistikkdager*, Nordgard.
17. S.H. Hashemi. 2000. Detecting Grammar Errors in Children's Writing: A Finite State Approach. In *Proceedings of the 13th Nordic Conference on Computational Linguistics (NoDaLiDa'01)*, Uppsala, Sweden.
18. J. Foster and O. E. Andersen. 2009. Generrate: Generating errors for use in grammatical error detection. In *Proceedings from the 4th Workshop on Innovative Use of NLP for Building Educational Applications.*
19. I. Aduriz, M. Aranzabe, J. M. Arriola, A. Díaz de Ilarraza, K. Gojenola, M. Oronoz, and L. Uria. 2004. A cascaded syntactic analyser for Basque. In *CicLing2004, Korea.*
20. N. Ezeiza, I. Aduriz, I. Alegria, J.M. Arriola, and R. Urizar. 1998. Combining stochastic and rule-based methods for disambiguation in agglutinative languages. In *COLING 1998*, Montreal.
21. K. Gojenola and K. Sarasola. 1994. Aplicación de la relajación gradual de restricciones para la detección y corrección de errores sintácticos. In *Actas de SEPLN94)*, Córdoba, Spain.
22. A. R. Golding and D. Roth. 1999. A Winnow-Based Approach to Context-Sensitive Spelling Correction. *Machine Learning*, 34(1-3):107–130.