Gonzalez-Dios, I., Aranzabe, M.J., Díaz de Ilarraza, A., Soraluze, A. (2013). *Detecting Apposition for Text Simplification in Basque*. In: Gelbukh, A. (eds) **Computational Linguistics and Intelligent Text Processing. CICLing 2013**. Lecture Notes in Computer Science, vol 7817. Springer, Berlin, Heidelberg. This version of the paper has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1007/978-3-642-37256-8_42

Detecting Apposition for Text Simplification in Basque

Itziar Gonzalez-Dios*, María Jesús Aranzabe,
Arantza Díaz de Ilarraza, and Ander Soraluze**

IXA NLP Group, University of the Basque Country (UPV/EHU),
Manuel Lardizabal 1 48014 Donostia
{itziar.gonzalezd,maxux.aranzabe,a.diazdeilarraza,ander.soraluze}@ehu.es
http://ixa.si.ehu.es/Ixa

Abstract. In this paper we have performed a study on Apposition in Basque and we have developed a tool to identify and to detect automatically these structures. In fact, it is necessary to detect and to code this structures for advanced NLP applications. In our case, we plan to use the Apposition Detector in our Automatic Text Simplification system. This Detector applies a grammar that has been created using the Constraint Grammar formalism. The grammar is based, among others, on morphological features and linguistic information obtained by a named entity recogniser. We present the evaluation of that grammar and moreover, based on a study on errors, we propose a method to improve the results. We also use a Mention Detection System and we combine our results with those obtained by the Mention Detector to improve the performance.

Keywords: Apposition Detector, Basque, Automatic Text Simplification, Mention Detection

1 Introduction

Automatic Text Simplification (TS) is a Natural Language Processing (NLP) task whose aim is to simplify texts automatically, keeping the meaning of original text, or at least avoiding information loss. TS is a necessary research line in NLP since the texts which are simplified are easier to process both for people and advanced NLP applications.

TS systems have already been proposed for people with disabilities [1], illiterate [2] or people who learn foreign languages [3] [4] among others. There are TS systems for advanced applications such us machine translation [5], Q&A systems [6], information extraction systems [7], and so on.

Our main motivation for TS is that long sentences cause problems in advanced applications like machine translation [8]. Apposition is a phenomenon that increases the length of the sentences and it has been reported in the context of TS as

* Itziar Gonzalez-Dios's work is funded by a PhD grant from the Basque Government.
** Ander Soraluze's work is funded by PhD grant from Euskara Errektoreordetza.

a complex phenomenon and rules to simplify these structures have been studied e.g. in [9] and [10] and for Basque in [11]. The information that an appositional phrase contains is not syntactically necessary and therefore it can be taken out of the sentence. This will mean the loss of some information, unless we create a new sentence out of the apposition. So if we remove apposition out of the sentence and create shorter sentences, for example, the task of machine translation will be more affordable.

In NLP, apposition detection has been mainly studied in the context of its integration in other general tools. However, there are tools that identify apposition explicitly [12] by means of machine learning techniques. Other techniques that have been used to detect apposition are heuristics [13] or full parse information [14]. In [15] appositive detection is applied as preprocess of a mention detection system and they use patterns to identify these structures. In [16] they use sequence mining to detect linguistic patterns in French like appositive qualifying phrases.

There are two tools in Basque that can be useful to detect Apposition. The first is a named entity recogniser and classifier, *Eihera* [17] and the second is the combination of the rule based (*IXAti* [18]) and the statistical-based (*ML-IXAti* [19]) shallow syntactic parsers for Basque. These tools consider apposition inside a noun phrase (restrictive) as a chunk, and apposition, that is expressed by noun phrase as appositive (non-restrictive), as more than an independent chunk. Since there is no explicit way to mark the apposition, we need a special tool to detect them.

So, in this paper we present a rule based Apposition Detector, based on linguistic knowledge, that is able to identify these structures and classify them according to their type. The output of this tool is human friendly, but it can be easily coded for machines as well. Although the first use of this Detector is TS, the Apposition Detector can be useful for other NLP advanced applications like mention detection, coreference resolution, parsing, textual entailment, text summarisation, Q&S systems, information extraction, event extraction, opinion mining etc. In the evaluation, we obtain 0.80 in F-measure. However, we analyse the errors and to improve the results, we use a Mention Detection System [20].

This paper is structured as follows: in section 2 we present the apposition types in Basque language. In section 3 the framework and the formalism of the Apposition Detector is explained. In section 4 we show the evaluation results. To improve this result we show in section 5 the experiments we carried out using the Mention Detector. In section 6 we describe how this tool will be used for Automatic Text Simplification and finally, in section 7 we expose the conclusion and the future work.

2 Apposition in Basque

Basque is Pre-Indo-European language and differs considerably in grammar from the languages spoken in surrounding regions. It is, indeed, an agglutinative head-final pro-drop isolated language whose case system is ergative-absolutive. Basque displays a rich inflectional morphology. Basque is still undergoing the normalisa-

tion process, and in charge of that, among others, there is *Euskaltzaindia* (Royal Academy of the Basque Language).

Apposition detection grammar has been built according to *Euskaltzaindia* [21]. As regulated, there are two types of apposition in Basque:

- **First type (restrictive):** Apposition that occurs inside a noun phrase. There are two ways to realise this type: a) example (1), the named entity *Luis Uranga* precedes the common name *presidenteak* (henceforth, type 1A):

(1) *Luis Uranga presidenteak (...)*
Luis Uranga president_the
'The president Luis Uranga (...)'

or b) example (2), the common name *presidente* precedes the named entity *Luis Uranga* (henceforth, type 1B):

(2) *Errealeko presidente Luis Uranga (...)*
Real_Sociedad_of president Luis Uranga
'The president of Real Sociedad Luis Uranga (...)'

- **Second type (non-restrictive):** A noun phrase as appositive like (3)[1]:

(3) *Jakinduria hori, guretzat harrapezina dena, (...)*
Wisdom that, us_for unattainable is_which_the,
'That wisdom, that is unattainable for us, (...) '

It is possible as well to combine both types (4):

(4) *Simon Peres laborista, Israelgo lehen ministro izana,*
Shimon Peres Labour_the, Israel_of Prime Minister have_been_the
'Labour Shimon Peres, the former Prime Minister of Israel, (...)'

and to merge the both structures (1A and 1B), example (5):

(5) *Vatikanoko Estatuekiko Harremanetarako idazkari Jean Louis Tauran artzapezpikuak (...)*
Vatican_of states_with relations_for secretary Jean Louis Tauran archbishop_the
'The archbishop Jean Louis Tauran, Secretary for Relations with States of The Vatican, (...)'

Parenthetical structures are not considered as apposition by *Euskaltzaindia*, since there is no agreement. However, some kind of parenthetical structures follow

[1] Notice that the equivalent translation is a relative clause

the same pattern as apposition in the simplification rules [11], so we have included rules to treat them in this grammar. For non simplification uses, these rules can be omitted. In (6) we see an example of a parenthetical structure the grammar covers.

(6) *Durangon (Bizkaia)*
Durango_in (Biscay)
'in Durango (Biscay)'

These are the target structures for our Apposition Detector. Each structure is given a tag, so they are classified.

If we applied only our shallow syntactic parser *IXAti* [18], type one apposition (both 1A and 1B) will be considered as a chunk, which is correct and valid for shallow parsing. But for some tasks like Automatic Text Simplification they should be distinguished. Apposition type two is considered by *IXAti* as more than one chunk. In both cases there is no explicit tag to express the appositional relation. This way Apposition Detector accomplishes this tagging task before the chunker *IXAti* is applied.

3 Architecture of the Apposition Detector

In this section we explain how our Apposition Detector works. Having as input a text, we perform the following analysis before we apply the Apposition Detector:

- **Morpho-syntactic analysis:** *Morpheus* [22] makes word segmentation and part of speech tagging. Syntactic function identification is made by *Constraint Grammar* formalism [23].
- **Lemmatisation and syntactic function identification:** *Eustagger* [24] resolves the ambiguity caused at the previous phase.
- **Multi-words items identification:** The aim is to determine which items of two or more words are always next to each other [25] [26].
- **Named entity recognition:** *Eihera* [17] identifies and classifies named-entities in the text (person, organisation, location).

To detect the apposition we have written a grammar following Constraint Grammar formalism [23]. The linguistic features we have used to write the rules in grammar are category, subcategory, and named entity tags.

Our detection system works in two phases: first, a grammar tags the named entities that are candidates to be a part of an apposition and secondly, based on the previous tags another grammar tags the second part of the apposition, if it fulfils the conditions of being a real apposition. The phrase with both tags is an apposition. There are 37 rules for the first phase, and 21 rules for the second phase. The rules are classified according to the entity type as well.

Each structure presented in section 2 has a tag (Table 1). This is the way apposition classification is made. This classification is valid, for example to know what kind of structures are used frequently or which rule should be applied for Text Simplification.

Table 1. Tags applied by the grammar

Type	1 appositional phrase	2 appositional phrases
1A	]APOS1	[APOS2
1B	]APOS1_KONTRA	[APOS2_KONTRA
2	]APOS1SINT	[APOS2SINT
Parenthetical structures	]APOS1_EGON	[APOS2_EGON

Once the apposition has been tagged we apply the rule based chunker *IXAti* [18] and *ML-IXAti* [19], which identifies chunks and clauses by combining rule-based grammars and machine learning techniques, exactly the version implemented in [20] to get the both appositional phrases. The algorithm is the following: the first appositional phrase begins where the chunker has tagged the phrase begin and it finishes with the word that has the first tag by our grammar. The second appositional phrase is formed by the word(s) between the first tag and second tag.

Let see this process with example (1), *Luis Uranga presidenteak*. The first rule (Figure 1) tags *]APOS1* and targets the end boundary of a named entity classified as person *Luis Uranga*, that is composed only by two words[2] and that is in the context of an apposition, in this example *Uranga*.

```
MAP (]APOS1) TARGET (ENTI_BUK_PER) IF (-1 ENTI_HAS_PER)  (0 NEXT_KM)
                                        (1  IZE + ARR) (NOT 1 NEXT_KM);
```

Fig. 1. CG rule to tag a candidate appositional phrase

The second rule (Figure 2) tags *[APOS2* and targets a common name, if previously an apposition candidate has been tagged (i.d. there is previously *]APOS1* tag), that is not followed by a adjective, in this example *presidenteak*.

```
MAP ([APOS2) TARGET  (IZE) IF (0C ARR) (NOT 0 PUNT_KARDI OR LEKU )
                                (NOT 0 HM OR DM) (-1 APO) (NOT 1 ADJ) ;
```

Fig. 2. CG rule to tag second appositional phrase and confirm the apposition

Taking into account the information of *IXAti* and *ML-IXAti* and the previously mentioned tags, the whole appositional phrases are *Luis Uranga* and *presidenteak*. In figure 3 we see the output of example (1) in text version (human-friendly).

[2] *ENTI_HAS_PER* and *ENTI_HAS_PER* tag the beginning and the ending of a named entity, and the other tags express morphological features.

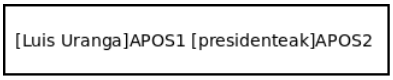

Fig. 3. Output of Apposition Detector in Text Version

4 Evaluation and error analysis

The corpus that has been used to develop and to evaluate the grammar has been EPEC (*Euskararen Prozesamendurako Erreferentzia Corpusa*-Reference Corpus for the Processing of Basque) [27]. EPEC Corpus is interesting for this task since it compiles text from newspapers, where apposition is a normal feature. In the first column of table 2 we see the quantities of the apposition found in the evaluation part of the corpus, in general and classified according to their type. To evaluate this grammar we have created a gold standard, where the apposition has been manually tagged.

In table 2 we also show the results[3] obtained by Apposition Detection and the quantities that are in the corpus. We show the results according to the apposition type as well.

Table 2. Evaluation results of the Apposition Detection

	Quantities	**Precision**	**Recall**	**F measure**
All types	336	0.87	0.74	0,80
1A type	286	0.90	0.62	0.73
1B type	30	0.85	0.73	0.79
2 type	9	1	0.44	0.62
Parenthetical structures	11	1	0.64	0.78

Except for a case, appositions were classified correctly. It was the case of a parenthetical structure that was considered as 1A type.

These results have been analysed qualitatively and we found out following errors and missing structures:

- Due to errors in named entity detection, rules were not applied or misapplied
- Apposition was detected, but a tag was not in the correct place. For example, the tag was in the substantive, when it should be in the adjective
- Complex appositional phrases that were already dismissed in development phase because they made a lot of errors for a correct one, like coordination in appositional phrases.

[3] Precision = correctly detected apposition/detected apposition; Recall = correctly detected apposition/all apposition; F-measure = 2 * precision * recall / (precision + recall)

5 Improving Apposition Detection using a Mention Detector

By analysing the results (section 4) we noticed that in some cases Apposition Detector has tagged the candidate (first tag) but due to the complexity of the appositional phrases, the tag for the second appositional phrase has been omitted (rule failed or dismissed rule) and in other cases nothing was retrieved. Those were considered as errors. This is the case of example (7).

(7) *Manuel Contreras Inteligentzia Nazionaleko Zuzendaritzako (DINA)*
Manuel Contreras Intelligence national_of direction_of (DINA)
buruzagi ohiak
head former

'Manuel Contreras, former head of the National Intelligence Directorate (DINA), '

In order to get this complex structures (e.g, (7)), we have carried out an experiment with the Mention Detector [20]. This system identifies mentions that are potential candidates to be part of coreference chains in Basque written texts. The aim of this experiment is to see if the Mention Detector can help to improve the results, without making changes in the system. In other words, we want to combine the output of the grammar and the output of the Mention Detector to see if we can get the discarded instances. This process is illustrated in figure 4.

We have formed two hypotheses that we explain next and developed a technique for each one. To test these hypotheses we made a subcorpus with the errors the grammar made, that is, we used the phrases which the first candidate was tagged, but the second one was omitted. There are 25 instances in this subcorpus. We only used 1A type, because other type quantities were insignificant.

Taking into account that this subcorpus was formed by the structures the grammar failed, we form **the first hypothesis**: If inside a mention is an appositional phrase candidate according to the grammar, it may be an apposition. So, the algorithm we implemented is next: if a mention has first tag inside (candidate), the rest of the mention is given second tag, and therefore considered as an apposition. Out of 25 instances 5 were were retrieved correctly.

To continue improving the results and taking into account the results of the first hypothesis, we formed **the second hypothesis**: if a mention is an appositional phrase candidate, the following mention in text should be its appositive. The technique we use to track is the mention identification number. If the candidate mention has identification number 1 in text, mention with identification number 2 should be its appositive. Applying this method, the 13 instances of the 20 left were correctly retrieved. Three instances more were retrieved, but as the whole appositional phrase was not correct, they were consider as errors.

So, we concluded that Mention Detection, without having been tuned, can improve the detection of apposition, retrieving 18 instances out of 25 and obtaining following results in the error subcorpus (Table 3). This approach using the

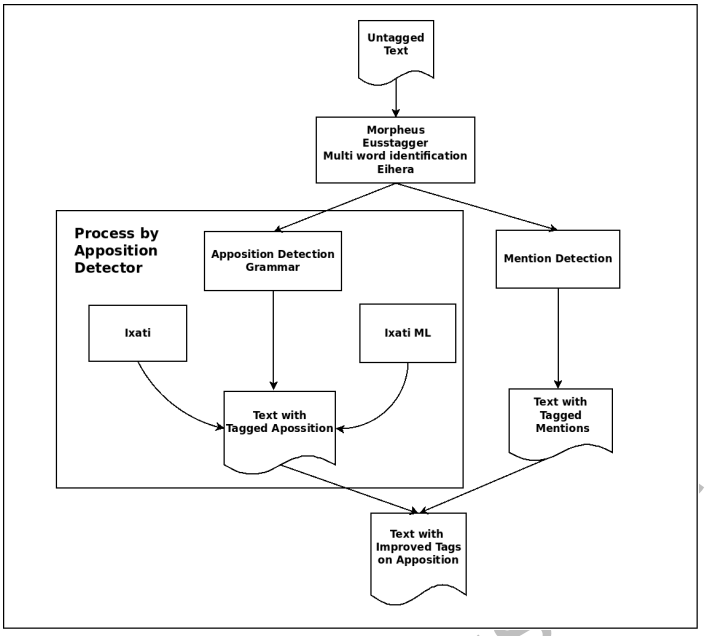

Fig. 4. Architecture of Apposition Detector and Improvement Process through Mention Detection

Mention Detector is above all helpful to retrieve the cases which grammarians had discarded the rule due to error increasing.

Table 3. Evaluation Results of Error Detection through Mention Detection

	Quantities	**Precision**	**Recall**	**F measure**
A1 type	25	0.86	0.72	0.78

It is important to mention that these algorithms have been tested with errors. To prove both hypotheses in a normal corpora, we think that the Mention Detector should be tuned. That is, instead of applying the second grammar, if we want to use only the Mention Detector, we should make severe changes. These algorithms should be more accurate, since not all the candidates form apposition. That is, we should incorporate the information of the second grammar adequated to the rules of the mention detection system, so that instances like named entities referring to a place followed by cardinal directions like *Londres mendebalean* (in West London) or followed by complex postpositions like *Erroma inguruan* (in the surroundings of Rome) are not retrieved. Anyway, we could not get rid of the grammar, since there are instances that Mention Detector would not retrieve.

6 Use of Apposition Detector in Text Simplification

The Apposition Detector presented here will be a part of the framework in our TS system, together with *Mugak* [28], the clause identifier. Based on its output apposition follows the simplification process [29], that will be explained by means of example (8): *Jasser Arafat buru palestinarra Egiptoko presidente Hosni Mubarak-ekin bildu zen atzo Kairon* (Palestinian Chairman Jasser Arafat met President of Egypt Hosni Mubarak yesterday in Cairo).

1. **Splitting:** First apposition is detected: there are two in sentence (9): [*Jasser Arafat buru palestinarra*] (Palestinian Chairman Jasser Arafat)and [*Egiptoko presidente Hosni Mubarak-ekin*] (President of Egypt Hosni Mubarak) . Secondly, a chunk is created for each appositional phrase in each apposition (figure 5). This is the task that the Apposition Detector presented in section 3 carries out.

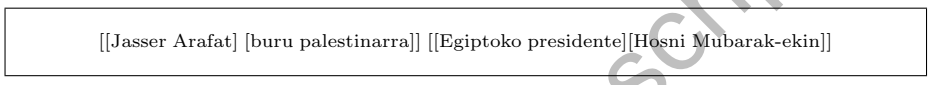

Fig. 5. Appositional phrases in sentence (8)

2. **Reconstruction:**
   (a) Removing: The chunks with the second tag (second appositional phrase) will be removed from the original sentence, obtaining following output: [*Jasser Arafat Hosni Mubarak-ekin bildu zen atzo Kairon*] (Jasser Arafat met Hosni Mubarak yesterday in Cairo). If a chunk has a suffix like *-ekin* (with) in *Hosni Mubarak-ekin* it should be removed.
   (b) Adding: Chunks with both tags will be added together with the copula, in these examples *da* (is), to form simple sentences: The absolutive suffix *-a* should be added in the phrase *Egiptoko presidentea* (the President of Egypt).

   This is the output of this operation: [*Jasser Arafat buru palestinarra da*] (Jasser Arafat is a Palestinian chairman) and [*Egiptoko presidentea Hosni Mubarak da*] (The president of Egypt is Hosni Mubarak). In this operation sentences have been created, but the simplification process is not yet fulfilled.
3. **Reordering:**
   (a) Internal word reordering in sentence: First the internal order will be checked: the order of former original sentence is kept untouched, the new sentences follow this rule pattern: Chunk with first tag (SUBJ), chunk with second tag (PRED), copula in present tense, 3 person, singular or plural depending on the subject. The first apposition follows the pattern of the rule, so it is left untouched but the second should be reordered to follow that

pattern[4]: [*Hosni Mubarak Egiptoko presidentea da*] (Hosni Mubarak is the president of Egypt).

(b) Sentence reordering in text: First, the former original sentence; then, new simple sentences following the order they appear in the original sentence.

4. **Correction:** There is no grammatical error to correct but sentences should be punctuated. This will be the final output: [*Jasser Arafat Hosni Mubarak-ekin bildu zen atzo Kairon. Jasser Arafat buru palestinarra da. Hosni Mubarak Egiptoko presidentea da.*] (Jasser Arafat met Hosni Mubarak yesterday in Cairo. Jasser Arafat is a Palestinian chairman. Hosni Mubarak is the President of Egypt.).

Following this process we have got shorter sentences which are useful for advanced applications like machine translation. Anyway, as simplification rules can be tuned according to the target audience, another option is to make a coordinate sentence with *eta* (and) to unify the new simple sentences. This will be the final output: *Jasser Arafat Hosni Mubarak-ekin bildu zen atzo Kairon. Jasser Arafat buru palestinarra da eta Hosni Mubarak Egiptoko presidentea da.* (Jasser Arafat met Hosni Mubarak yesterday in Cairo. Jasser Arafat is a Palestinian chairman and Hosni Mubarak is the President of Egypt.).

7 Conclusion and Future work

In this paper we have presented an Apposition Detector based on linguistic knowledge. Moreover, it is able to classify the apposition in corpora according to their type and structure, which is helpful for linguistic analysis and research on apposition.

We have evaluated this tool and looking at the results (F-measure 0.80), we realised that they could be improved. So we have made an experiment on errors with another tool, the Mention Detector. We have formed two hypotheses and created to techniques to combine the output of the grammar and the output of the Mention Detector. This way, the instances that were not covered by the grammar were retrieved (F-Measure 0.78), without having changed the Mention Detection system.

We have explained as well how we are going to use the output of the Mention Detector in Automatic Text Simplification by means of an example. Performing the syntactic simplification process, we get shorter sentences that are easier to process for NLP advanced applications such us machine translation.

Although the first use of the Apposition Detector is Automatic Text Simplification, it can be used for other tasks like coreference resolution, information extraction, lexicon elaboration or text summarisation. Indeed, we plan to implement this Detector to improve the mention detection system and in the coreference resolution system.

[4] Before reordering this sentence was already grammatically correct, since Basque is a free word order language. But according to the simplification rule, the order should change.

Acknowledgments. Itziar Gonzalez-Dios's work is funded by a PhD grant from the Basque Government and Ander Soraluze's work is funded by PhD grant from Euskara Errektoreordetza, the University of the Basque Country (UPV/EHU). This research was also supported by the the Basque Government (IT344-10).

References

1. Carroll, J., Minnen, G., Pearce, D., Canning, Y., Devlin, S., Tait, J.: Simplifying Text for Language-Impaired Readers. In: 9th Conference of the European Chapter of the Association for Computational Linguistics. (1999)
2. Candido, Jr., A., Maziero, E., Gasperin, C., Pardo, T.A.S., Specia, L., Aluisio, S.M.: Supporting the adaptation of texts for poor literacy readers: a text simplification editor for Brazilian Portuguese. In: Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications. EdAppsNLP '09, Stroudsburg, PA, USA, Association for Computational Linguistics (2009) 34–42
3. Petersen, S.E., Ostendorf, M.: Text Simplification for Language Learners: A Corpus Analysis. Electrical Engineering (SLaTE) (2007) 69–72
4. Burstein, J.: Opportunities for Natural Language Processing Research in Education. In: Computational Linguistics and Intelligent Text Processing. Springer Berlin / Heidelberg (2009)
5. Poornima, C., Dhanalakshmi, V., Anand, K., Soman, K.: Rule based Sentence Simplification for English to Tamil Machine Translation System. International Journal of Computer Applications **25**(8) (2011) 38–42
6. Bernhard, D., De Viron, L., Moriceau, V., Tannier, X.: Question Generation for French: Collating Parsers and Paraphrasing Questions. Dialogue and Discourse **3**(2) (2012) 43–74
7. Jonnalagadda, S., Gonzalez, G.: Sentence simplification aids protein-protein interaction extraction. Arxiv preprint arXiv:1001.4273 (2010)
8. Labaka, G.: EUSMT: Incorporating Linguistic Information into SMT for a Morphologically Rich Language. Its use in SMT-RBMT-EBMT hybridation. PhD thesis, UPV-EHU (2010)
9. Siddharthan, A.: Syntactic simplification and text cohesion. Research on Language & Computation **4**(1) (2006) 77–109
10. Specia, L., Aluisio, S.M., Pardo, T.A.: Manual de Simplificação Sintática para o Português. Technical Report NILC-TR-08-06, So Carlos-SP. (2008)
11. Gonzalez-Dios, I.: Euskarazko egitura sintaktikoen azterketa testuen sinplifikazio automatikorako: Aposizioak, erlatibozko perpausak eta denborazko perpausak. Master's thesis, University of the Basque Country (September 2011)
12. Freitas, M.C., Duarte, J.C.N., Santos, C., Milidiú, R.L., Rentería, R.P., Quental, V.: A machine learning approach to the identification of appositives. Advances in Artificial Intelligence-IBERAMIA-SBIA 2006 (2006) 309–318
13. Phillips, W., Riloff, E.: Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In: Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, Association for Computational Linguistics (2002) 125–132
14. Roth, D., Sammons, M.: Semantic and logical inference model for textual entailment. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics (2007) 107–112

15. Kummerfeld, J.K., Bansal, M., Burkett, D., Klein, D.: Mention detection: heuristics for the OntoNotes annotations. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task. CONLL Shared Task '11, Stroudsburg, PA, USA, ACL (2011) 102–106
16. Béchet, N., Cellier, P., Charnois, T., Crémilleux, B.: Discovering linguistic patterns using sequence mining. In: Proceedings of the 13th international conference on Computational Linguistics and Intelligent Text Processing - Volume Part I. CICLing'12, Berlin, Heidelberg (2012) 154–165
17. Fernandez Gonzalez, I.: Euskarazko Entitate-Izenak: identifikazioa, sailkapena, itzulpena eta desanbiguazioa. PhD thesis, UPV-EHU (2012)
18. Aduriz, I., Aranzabe, M.J., Arriola, J.M., Díaz de Ilarraza, A., Gojenola, K., Oronoz, M., Uria, L.: A cascaded syntactic analyser for basque. Computational Linguistics and Intelligent Text Processing (2004) 124–134
19. Arrieta, B.: Azaleko sintaxiaren tratamendua ikasketa automatikoko tekniken bidez: euskarako kateen eta perpausen identifikazioa eta bere erabilera koma-zuzentzaile batean. PhD thesis, UPV-EHU (2010)
20. Soraluze, A., Arregi, O., Arregi, X., Ceberio, K., Díaz de Ilarraza, A.: Mention Detection: First Steps in the Development of a Basque Coreference Resolution System. In: Proceedings of KONVENS 2012. (2012) 128–163
21. Euskaltzaindia: Euskal gramatika laburra: perpaus bakuna. Euskaltzaindia (2002)
22. Alegria, I., Aranzabe, M.J., Ezeiza, A., Ezeiza, N., Urizar, R.: Robustness and customisation in an analyser/lemmatiser for Basque. In: LREC-2002 Customizing knowledge in NLP applications workshop. (2002) 1–6
23. Karlsson, F., Voutilainen, A., Heikkila, J., Anttila, A.: Constraint Grammar, A Language-independent System for Parsing Unrestricted Text. Mouton de Gruyter (1995)
24. Aduriz, I., Aldezabal, I., naki Alegria, I., Arriola, J.M., de Ilarraza, A.D., Ezeiza, N., Gojenola, K.: Finite State Applications for Basque. In: EACL'2003 Workshop on Finite-State Methods in Natural Language Processing. (2003) 3–11
25. Ezeiza, N.: Corpusak ustiatzeko tresna linguistikoak. Euskararen etiketatzaile morfosintaktiko sendo eta malgua. PhD thesis, UPV-EHU (2002)
26. Urizar, R.: Euskal lokuzioen tratamendu konputazionala. PhD thesis, UPV-EHU. (2012)
27. Aduriz, I., Aranzabe, M.J., Arriola, J.M., Atutxa, A., Díaz de Ilarraza, A., Ezeiza, N., Gojenola, K., Oronoz, M., Soroa, A., Urizar, R. In: Methodology and steps towards the construction of EPEC, a corpus of written Basque tagged at morphological and syntactic levels for automatic processing. Volume 56. Rodopi (2006) 1–15
28. Aranzabe, M.J., Díaz de Ilarraza, A., Gonzalez-Dios, I.: Transforming Complex Sentences using Dependency Trees for Automatic Text Simplification in Basque. Manuscript
29. Aranzabe, M.J., Díaz de Ilarraza, A., Gonzalez-Dios, I.: First Approach to Automatic Text Simplification in Basque. In Rello, L., Saggion, H., eds.: Proceedings of the Natural Language Processing for Improving Textual Accessibility (NLP4ITA) workshop (LREC 2012), Istanbul, Turkey (2012) 1–8