Example-Based Treebank Querying with GrETEL – now also for Spoken Dutch

Liesbeth Augustinus, Vincent Vandeghinste, Ineke Schuurman, and Frank Van Eynde

Centre for Computational Linguistics, University of Leuven

`{liesbeth,vincent,ineke,frank}@ccl.kuleuven.be`

ABSTRACT

Although several syntactically annotated corpora (or treebanks) exist for Dutch, they are seldomly used for descriptive linguistic research because there are no easy-to-use exploitation tools available. This demonstration paper describes *GrETEL*, a linguistic search engine (`http://nederbooms.ccl.kuleuven.be/eng/gretel`) that enables non-technical users to consult treebanks in a user-friendly way. Instead of a formal search expression, a natural language example is used as input to the system, allowing users to search for similar constructions as the example they provide. In the first version of GrETEL, only written Dutch (LASSY) was included. Based on user requests we have now included the Spoken Dutch Corpus (CGN) as well.

KEYWORDS: Dutch, treebank, querying, example-based.

1 Introduction

Within CLARIN (**C**ommon **L**anguage **R**esources and Technology **In**frastructure), a European research infrastructure project, researchers in language and speech technology aim at making it easy for researchers in the humanities and social sciences to work with digital language data.[1] For linguists those language data could be annotated corpora, such as treebanks. While several syntactically annotated corpora (or treebanks) exist for Dutch, they are seldomly used for descriptive linguistic research up till now.

Talking with (descriptive) linguists about using treebanks, the same problems pop up over and over again: on the one hand the limited user-friendliness of the query languages and search tools, and on the other hand the lack of standardisation in both treebanks and query languages. This is unfortunate, since treebanks could be very useful for them, especially when they are looking for (possibly) discontinuous constructions. In order to overcome the querying problems, the linguistic search engine GrETEL (**Gr**eedy **E**xtraction of **T**rees for **E**mpirical **L**inguistics) was developed.[2] GrETEL was created as a part of the Nederbooms project, a CLARIN project funded by the Flemish Community.

Instead of developing yet another query language or designing yet another GUI, we present a query engine which does not ask for any formal input query. As input, the tool takes something linguists are familiar with: natural language. Since linguists tend to start their research from example sentences, the methodology of example-based querying allows users to search for similar constructions as the example they provide. How similar is for the user to decide.

A first version of GrETEL (1.0), which is optimized for the Dutch LASSY treebank (van Noord et al., 2013), is described in Augustinus et al. (2012). In a more refined version (GrETEL 1.1), the Spoken Dutch Corpus (CGN) (Oostdijk et al., 2002) is also supported. For the next release (GrETEL 2.0), we will make the tool less dependent on particular XML formats. Moreover, we will enable much larger treebanks to be queried compared to the ones that are supported at the moment.

2 Methodology and Design

As an example, we look for collective noun constructions, such as *een aantal mensen* 'a number of people' and *een school vissen* 'a school of fish'. In Dutch, such constructions consist of a determiner (e.g. *een* 'a'), a noun denoting the kind of collection (e.g. *aantal* 'number'), and a noun denoting the entities in the collection (e.g. *mensen* 'people'). As we want to find discontinous examples as well, such as *een school kleine vissen* 'a school of small fish', a treebank is the obvious resource to use, rather than a corpus just annotated for part of speech. The construction in mind may also determine which treebank is consulted. The LASSY treebank contains written Dutch, whereas the CGN treebank contains (transcribed) spoken language. For this example, we will use CGN, as case studies on the LASSY treebank were already described in previous work (Augustinus et al., 2012). We will furthermore indicate the differences between the updated version of GrETEL and the previous release.

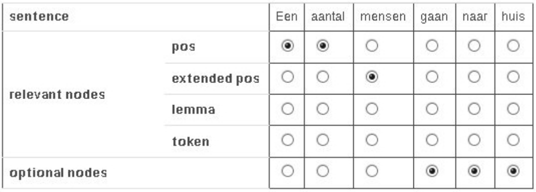
Work related to our approach is the now deceased Linguist's Search Engine (Resnik and Elkiss, 2005), a tool that also used example-based querying; and the TIGER Corpus Navigator (Hellmann et al., 2010), which is a SemanticWeb system used to classify and retrieve sentences from the TIGER corpus on the basis of abstract linguistic concepts.

[1] http://www.clarin.eu
[2] http://nederbooms.ccl.kuleuven.be/eng/gretel

Natural language example The user provides a relevant natural language example, containing the syntactic construction (s)he is looking for. After presenting an input construction (in this case: *een aantal mensen gaan naar huis* ‘a number of people go home’) to the system, GrETEL returns the sentence to the user in the *sentence parts selection matrix*, cf. Figure 1.

Selection The user can indicate for each word whether (s)he is interested in the short part of speech (pos), the extended pos, the lemma or the token.[3] For general non-lexical similarities, **pos** should be selected. To take into account more detailed information, such as agreement or number, **extended pos** should be selected. The **lemma** button should be indicated to abstract over word forms, and the **token** button should be selected for retrieving specific word forms. Mind that token is a case sensitive feature, while lemma is case insensitive (except for proper nouns).

If the input contains words that are not part of the target construction, the **optional nodes** button should be used. Note that both the dependency relation and the syntactic category of all relevant nodes are taken into account,[4] cf. the XPath expression (1) describing the subtree depicted in Figure 2b. Instead of marking certain words as optional, one could also use a sentence part as input (such as *een aantal mensen*).



sentence		Een	aantal	mensen	gaan	naar	huis
relevant nodes	pos	●	●	○	○	○	○
	extended pos	○	○	●	○	○	○
	lemma	○	○	○	○	○	○
	token	○	○	○	○	○	○
optional nodes		○	○	○	●	●	●

Figure 1: Sentence Parts Selection Matrix

We indicated **pos** for both *een* ‘a’ and *aantal* ‘number’, since we are looking for any determiner, followed by a noun. For *mensen* ‘people’, we indicated **extended pos**, as this wil return matches not merely with the same part of speech, but also with the same number (we are looking for plural nouns).

XPath expression GrETEL turns the information from the matrix into an XPath expression[5] (1), which can be used to query the treebank. In addition, the parse tree of the input sentence is presented, along with the subtree containing the construction the user is looking for, cf. Figure 2. Users can optionally adapt the XPath query in order to refine or generalize the search instruction. The user also has the option to search a complete treebank or to select one or more subcorpora. In the case of CGN, one could for example only query the Flemish part of the corpus.

[3] The short pos tags are different in LASSY and CGN. The LASSY treebank contains both the tags assigned by the Alpino parser (van Noord, 2006) (e.g. *noun, verb*) and the (extended) CGN/D-COI tags (e.g. N(soort,mv,basis), WW(pv,tgw,ev)) (Van Eynde, 2004). CGN only contains the CGN/D-COI tags, so the shortened versions of the CGN/D-COI tags (e.g. *n, ww*) are used as pos tags.

[4] The list of all dependency relations and syntactic categories can be found in Hoekstra et al. (2003) for CGN and in van Noord et al. (2011) for LASSY.

[5] `http://www.w3.org/TR/xpath`

(1)
```
//node[@cat="np" and node[@rel="det" and @cat="np" and node[@rel="det" and
@pt="lid"] and node[@rel="hd" and @pt="n"]] and node[@rel="hd" and @pt="n" and
@postag="N(soort,mv,basis)"]]
```

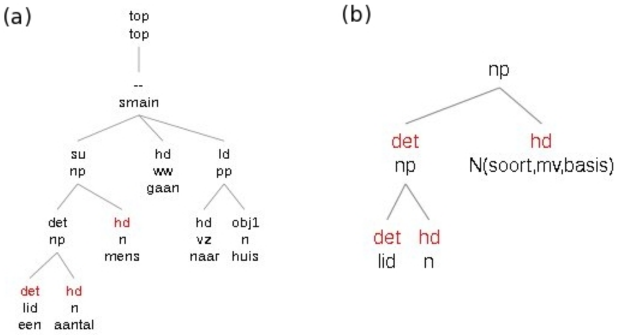


Figure 2: Parse tree (a) and subtree (b) of the input example

Results In the last phase the search results are presented, i.e. the sentences containing the construction at hand. The user can inspect the tree and/or or the source XML of all results. It is also possible to download the results in text format. For the query in (1), we found 594 matches in the complete CGN treebank (130k sentences, 1M tokens). Some examples are given in (2). Note that 2b is discontinuous.

(2) a. heb je gezien dat in de frigo dat er daar **een pakje** **pralines** stond?
have you seen that in the fridge that there there a packet-DIM chocolats stood
'Did you see there was a box of chocolats in the fridge?' [fva400282__2]

b. ik heb hier nog **een hele** **reut ouwe boeken** staan.
I have here still a whole lot old books stand
'I still have a whole lot of old books here.' [fnc008006__344]

Search options Besides adding the extended pos option to the *selection matrix*, we added some other options in GrETEL 1.1. It is now possible to include some context in the search results, to ignore the syntactic properties of the dominating node in the subtree, and to split up the extended pos tags. Due to limitations of space, we are not able to discuss the options here.

3 Technical description

GrETEL is accessible online, which means that users do not have to install any treebanks or specific software (e.g. a parser) locally.[6] Drupal is used as content management system.[7] Figure 3 presents GrETEL's general architecture, which consists of three tiers: the presentation layer, the search layer, and the data layer.

Presentation tier In the first tier we interact with the user, resulting in an information flow to and from the search layer. This is done using PHP and HTML.

[6] The user is adviced to access the tool via Mozilla Firefox (`http://www.mozilla.org`), as that browser supports W3C standards very well.

[7] `http://drupal.org`

Search tier This tier is used to process information coming from the presentation layer, and to interact with the data layer. The (Alpino) parser (van Noord, 2006) is addressed using PHP, while the *Subtree Finder* and *XPath Generator* are implemented as perl scripts. For the treebank search PHP, SQL, and XPath are used.

Data tier In the current implementation (cf. 'version 1' in Figure 3) the data is stored into a PostgreSQL database.[8] The data format of the treebanks is Alpino XML.[9]

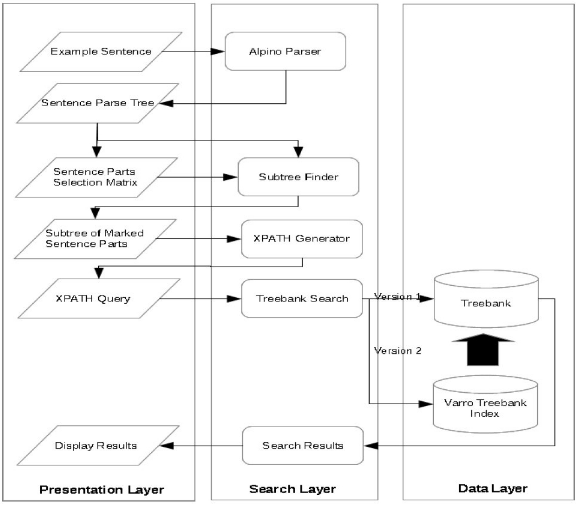


Figure 3: GrETEL's 3-tier architecture

4 Conclusion and Future Work

Thanks to the positive response to GrETEL 1.0, we can now implement GrETEL 2.0, which will be able to deal with larger treebanks, and which will also be able to deal with XML formats other than Alpino XML.

We plan to make the various layers less intertwined to make adaptation for other languages, treebanks, and/or data formats possible.

The major challenge, however, is making the system usable for very large treebanks, such as the LASSY large corpus (500M tokens). In the next implementation the PostgreSQL database will be replaced by a native XML database, such as BaseX,[10] and we will make use of the *Varro Treebank Indexer*, cf. 'version 2' in Figure 3.

Varro (Martens and Vandeghinste, 2010), (Martens, 2011) is a toolkit and algorithm for indexing regular structures in treebanks. Subtrees are listed in the Varro Treebank Index, which indicates where they are located and what their frequency is.

With this preprocessing step we intend to allow for faster treebank mining and, we hope, to query large (huge) treebanks in a timespan acceptable for the intended users, i.e. descriptive linguists.

[8] http://www.postgresql.org/

[9] http://www.let.rug.nl/vannoord/Lassy/alpino_ds.dtd

[10] http://basex.org

References

Augustinus, L., Vandeghinste, V., and Van Eynde, F. (2012). Example-Based Treebank Querying. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul.

Hellmann, S., Unbehauen, J., Chiarcos, C., and Ngomo, A.-C. N. (2010). The TIGER Corpus Navigator. In *Proceedings of TLT-9*, pages 91–102, Tartu, Estonia.

Hoekstra, H., Moortgat, M., Renmans, B., Schouppe, M., Schuurman, I., and van der Wouden, T. (2003). *CGN Syntactische Annotatie*. http://nederbooms.ccl.kuleuven.be/documentation/sa-man_cgn.pdf.

Martens, S. (2011). *Quantifying Linguistic Regularity*. PhD thesis, KU Leuven, Leuven, Belgium.

Martens, S. and Vandeghinste, V. (2010). An Efficient, Generic Approach to Extracting Multi-Word Expressions from Dependency Trees. In *Proceedings of the CoLing Workshop: Multiword Expressions: From Theory to Applications (MWE 2010)*, Beijing, China.

Oostdijk, N., Goedertier, W., Van Eynde, F., Boves, L., Martens, J.-P., Moortgat, M., and Baayen, H. (2002). Experiences from the Spoken Dutch Corpus Project. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC'02)*, pages 340–347, Las Palmas, Spain.

Resnik, P. and Elkiss, A. (2005). The Linguist's Search Engine: An Overview. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 33–36, Ann Arbor.

Van Eynde, F. (2004). *Part of Speech Tagging en Lemmatisering van het Corpus Gesproken Nederlands*. www.ccl.kuleuven.be/Papers/POSmanual_febr2004.pdf.

van Noord, G. (2006). At Last Parsing Is Now Operational. In *TALN 2006*, pages 20–42.

van Noord, G., Bouma, G., Van Eynde, F., de Kok, D., van der Linde, J., Schuurman, I., Tjong Kim Sang, E., and Vandeghinste, V. (2013). Large Scale Syntactic Annotation of Written Dutch: Lassy. In *Essential Speech and Language Technology for Dutch: Resources, Tools and Applications*. Springer.

van Noord, G., Schuurman, I., and Bouma, G. (2011). *Lassy Syntactische Annotatie, Revision 19455*. www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf.