

SABeD application manual

Table of contents

Introduction	3
Information about the corpus	3
Linguistics Annotation	3
Metadata categories	3
Subject area	3
Application user manual	4
Getting started	4
Searching the corpus	5
Simple search	5
Search	5
Wildcards	5
Reset	6
History	6
Global settings	6
Extended search	7
Basics	7
More annotations	8
Upload a list of values	8
Part of Speech dialogue box	8
Within	9
Batch Splitting	9
Filter search by	9
Advanced search	10
The query builder	10
The tab Search	10
Token attributes	11
Adding attributes to a token box	11
Function of the two +-buttons in a token box	11
The tab Context	12
Managing sequences of token boxes	12
Uploading value lists in the query builder	13
Within	13
Copy to CQL editor	13
Expert search	13
Copy to CQL editor	14
Import query	14
Gap filling	14
Viewing results	16
Per Hit view	16

Sorting results	16
Grouping results	17
Per Document view	18
Sorting results	18
Grouping results	19
Exporting results	19
Information about a document	19
Content	19
Metadata	20
Statistics	20
Exploring the corpus	20
Documents	20
N-grams	21
Options	21
Example	21
Statistics (frequency lists)	21
Options	22
Example	22
Appendix: Corpus Query Language	23
CQL support	23
Supported features	23
Differences from CWB	24
(Currently) unsupported features	24
Using Corpus Query Language	25
Matching tokens	25
Sequences	26
Regular expression operators on tokens	26
Punctuation	26
Case- and diacritics-sensitivity	27
Matching XML elements	27
Labeling tokens, capturing groups	28
Global constraints	28

Introduction

This manual describes the corpus exploitation environment of the Spoken Academic Belgian Dutch Corpus (SABeD). The corpus application is developed by the Dutch Language Institute (Instituut voor de Nederlandse Taal or INT). The backend of the application is the BlackLab Lucene based search engine developed for corpora with token-based annotation (<https://blacklab.ivdnt.org/>). The web-based frontend is a further development of the BlackLab Frontend application developed by INT (<https://github.com/instituutnederlandsetaal/blacklab-frontend>) in CLARIN and CLARIAH projects. Its design is inspired by the first version of the OpenSoNaR user interface by Tilburg and Radboud University (<https://github.com/Taalmonsters/WhiteLab2.0>).

Information about the corpus

The Spoken Academic Belgian Dutch Corpus consists of 200 lectures given in higher education institutions in Flanders. The first 25 and the last 5 minutes of each lecture were transcribed using an ASR system tuned to Belgian Dutch and then manual utterance segmentation was applied, followed by manual correction of the automated transcription. More information about the project can be found at: <https://www.arts.kuleuven.be/ling/language-education-society/projects/sabed>.

Linguistics Annotation

The resulting text has been processed with the Ucto tokenizer (<https://languagemachines.github.io/ucto/>) and the Frog (<https://languagemachines.github.io/frog/>) language processing (NLP) modules developed for Dutch. Part of Speech tagging uses the D-COI tagset which is an extension of the CGN tagset.

Documentation for the CGN tagset and the extended version used for written Dutch can be found in:

- Frank Van Eynde (2004). ["Part of Speech Tagging en Lemmatisering van het Corpus Gesproken Nederlands" \[English version\]](#)
- Frank Van Eynde (2005). ["Part of Speech Tagging en Lemmatisering van het D-Coi Corpus"](#) (slightly extended version of CGN tag set).

Metadata categories

SABeD has been enriched with the metadata category Subject area. This metadata category is described below. In the corpus application it is possible to limit a search by filtering on this metadata category.

Subject area

There are four Subject areas: Biology, Exact sciences, Humanities and Social sciences.

Application user manual

The language of the corpus application is set to Dutch by default. Press the globe icon in the top right corner to select English.

Getting started

Here are a few examples of what you can do with the corpus application (the links will take you to the application):

- To search for a word literally in the form you specify, use Simple search or Extended search.
 - Simple Search for Word [land](#)
 - Extended Search for Word [land](#)
- To search for different spellings and inflections/conversions of a given lemma, use Lemma in Simple Search or Extended Search.
 - Extended Search for Lemma [land](#)
- To search for words or lemmata satisfying a certain pattern, use *wildcards* in Simple Search or Extended Search, or *regular expressions* in Advanced Search and Expert Search.
 - words starting with *ver* and ending with *len* in [Simple Search](#)
 - lemmata starting with *ver* and ending with *len* in [Extended Search](#)
 - lemmata starting with *ver* and ending in *eren* with (mostly) one syllable in between in [Advanced Search](#)
 - lemmata starting with *ver* and ending in *eren* with (mostly) one syllable in between in [Expert Search](#)
- To search for a multiword pattern, e.g. all adjectives appearing before a given lemma as a singular noun, use the query builder in Advanced Search or use Expert Search.
 - lemma is *vraag* and is being followed by a form of the verb *zijn* in [Advanced Search](#)
- To see which unique forms occur as a result of your search, use Group Results.
 - example Group by Annotation: [different words following the word *echte*](#)
 - example Group by Annotation: [different words preceding the lemma *geval*](#)
- To explore the distribution of document properties in the corpus, use the Explore feature.
 - example: [characteristics of the Subject areas](#)

Searching the corpus

Simple search

Search

The Simple Search allows you to quickly search for specific word forms (e.g. *vroeger*). After entering a search term, a running list appears, which contains suggestions for possible search terms in alphabetical order, based on the characters typed in.

It is also possible to enter a phrase: *met de auto* or *ik ben geïnteresseerd*. You will then find all occurrences of that exact phrase. To start the search simply press enter or press the Search button. In the search field Word you can also search for different values simultaneously by separating them without spaces by a vertical line, e.g. *Jan|Piet|Emma*.

Note that in Simple Search the patterns will be matched case-insensitively: *ik* will deliver the same results as *Ik*. See the paragraph [Grouping results](#) in Per Hit view to see how it is nevertheless possible to distinguish between uppercase and lowercase letters, or go to Extended Search.

Wildcards

In Simple Search, the use of wildcards can prove good service to search for specific word forms or lemmata. A wildcard is a symbol used to replace or represent one or more characters. The following two wildcards are supported:

- * The asterisk matches any character zero or more times. Therefore, searching for *a*n* matches all word forms that start with an *a* and end with a *n*, e.g. *aan*, *alleen*, *androgenen* and *aandoeningen*.
- ? The question mark matches a single character once. Therefore, searching for *be?* matches *only* three-letter word forms or lemmata starting with an *be*, e.g. *ben*, *bed*, *beu* and *bek*.

This wildcard can be used more than once. Thus *d???n* matches *delen*, *dagen*, *duwen* and *d-gen*.

Note that searching with wildcards is limited to Simple Search and Extended Search. [In Advanced Search and Expert Search you can use so-called regular expressions instead of wildcards.]

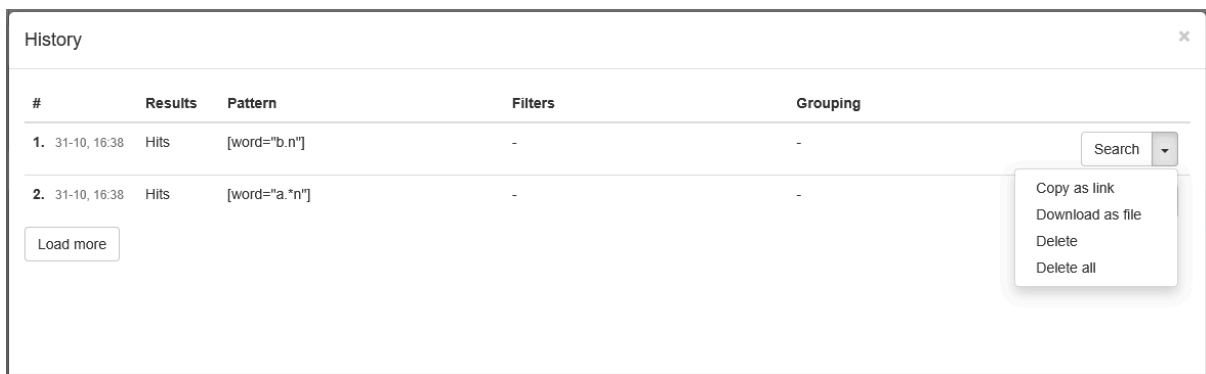
Reset

You can start a new search by pressing the Reset button. By doing so, both the search query and the hits found will be cleared. Your search history, however, will remain unchanged.

Note that it is also possible to start a new search by entering a new word or phrase in the search field.

History

The History button will display your query history. Per search query there are several possibilities (as shown in the screenshot below): you can perform the search query again (Search), you can copy the search query as a link (Copy as link), you can download the search query as a file (Download as file), you can delete a single search query (Delete) or delete all search queries (Delete all).



Every search query has its own url. If you copy this url via History (Copy as link) or directly from the address bar of your browser, you can send it to someone else who can import this link via Import from a link. It offers that person the possibility to run the search on his own computer.

Global settings

The Global settings dialogue, activated by pressing the wheel button, allows you to configure five settings: Results per page, Sample size, Seed, Context size and Wide View.

- *Results per page*: you can choose whether you want 20, 50, 100 or 200 results to be shown;
- *Sample size*: selecting a value here will instruct the search engine to return a random sample drawn from the complete result set. The sample size can be limited by
 - a percentage of the total number of search results (percentage);
 - the number of results displayed (count).
- *Seed*: a 'random seed' is a number used to initialize a so-called pseudo-random number generator. Keeping the same seed will ensure that two samples drawn from the same result set are identical. A new seed will most likely result in a different sample;
- *Context size*: by entering a number you can determine the number of tokens (words or punctuation marks) Before hit and After hit;
- *Wide View*: the default setting is 'small view'; you can change to Wide View by ticking the checkbox.

Global settings
✕

Results per page: 20 results per page ▾

Sample size: percentage ▾ Sample size ▾

Seed: Seed ▾

Context size: Context size ▾

Wide View

Close

Extended search

Like in Simple search, Extended Search allows you to quickly search for specific word forms or phrases. The search is performed in the same way as described for Simple Search. There are two tabs here: basics and more annotations.

Basics

In this corpus the three main attributes you can search for are Word (more precise: word form), Lemma and Part of Speech. In the search field Word and Lemma enter the word(s) or lemma(ta) (or Upload a list of values) you are looking for. Like in Simple Search, you can also enter a phrase here. In the search field Part of Speech you can select the desired values. All supported attributes are shown in the search form:

Simple
Extended
Advanced
Expert

Basics
More annotations

Word Word ⬆

Case- and diacritics-sensitive

Lemma Lemma ⬆

Case- and diacritics-sensitive

Part of Speech Part of Speech ✎

Within: Document Sentence

Split batch queries

In Extended Search it is also possible to search case- and diacritics-sensitive. Note that the default setting for search is case- and diacritics-insensitive. For example, searching for the Word *Jan* will result in 106 occurrences. By ticking the box Case- and diacritics-sensitive you will find 0 occurrences of the Word *jan*, but 106 of *Jan*.

Like in Simple Search, wildcards are supported in Extended Search. (See for a short explanation of wildcards [Simple Search](#).)

You can combine multiple fields (such as Word and Part of Speech) to form a single query. For example, searching for the Word (form) *was* together with the Part of Speech VRB will result in a list of all verbs containing the word form *was*. The syntax of your query is shown in the results: [\[word="was"&pos_head="vw"\]](#).

More annotations

You can expand your search options by using the tab More annotations. The following screen will appear:

The screenshot shows the search interface with the 'More annotations' tab selected. The search bar is empty. The filter section is set to 'Subject area'. The search options are set to 'Simple', 'Extended', 'Advanced', and 'Expert'. The search options are set to 'Basics' and 'More annotations'. The search options are set to 'Adposition Type', 'Article Type', 'Case', 'Conjunction Type', 'Degree', 'Dialect', 'Gender', and 'Inflection'. The search options are set to 'Document' and 'Sentence'. The search options are set to 'Split batch queries'.

All the values of these attributes can be selected by the use of a drop-down menu.

Upload a list of values

At the right side of the search fields Word and Lemma there is an option to Upload a list of values; those values must all be separated by a white space. Note that this function only works for *.txt-files. (If you are using a text editor like Word, you have to save your file as a *.txt-file first.)

Every word in the uploaded file will be added to the list of values to search for. To remove the word list simply delete all text in the search field or press the Reset button.

Part of Speech dialogue box

Clicking on the pencil next to the search field Part of Speech provides you with the Part of Speech dialogue box.

Part of Speech

Adjective
Adposition
Adverb
Article
Conjunction
Interjection
Noun
Numeral
Pronoun
Special
Verb
Punctuation

Within

It is possible to apply your search query to all documents of the corpus, which is the default setting. However, it may be desirable to limit a search to a document, a sentence or an utterance. The main reason to do this is to exclude multi-word matches that stretch over a sentence boundary. To do so you can click on these respective buttons.

Batch Splitting

Ticking the option Split batch queries will split a query containing a certain number of attributes, separated by the character | or the boolean operator OR, in a set of smaller queries equal to the number of chosen attributes. Thus every value will be executed as a separate query. The metadata filters filled in by Filter search by (see below) will apply to all subqueries.

After running the query, only the hits for the *last* value will be shown in the results. To obtain the results for the other values you have to go to History, select the value (i.e. pattern) you are looking for and press Search on the right hand side of the history panel.

An example will make this clear. Searching for *biologie|techniek* in Lemma will show all hits for *biologie* and *techniek*. When the option Split batch queries is ticked on, the same query will show only hits of *techniek* (being the last value). In History you will therefore find *techniek* on top, followed by the query *biologie*.

Filter search by

At the right side you will find the option to limit your query to a subset of documents with a specific metadata value for Subject area. To view the results for all documents, simply leave the attributes in the filtering form empty. Subject area allows you to choose one or more values from a drop-down list.

You can pick one of these values by clicking on it; your choice will be marked with a tick. It is possible to choose several values. If you want to delete a selection, you can click on the corresponding line again. To close the drop-down list, you can either press the upward pointing arrow in the upper right corner or simply press escape.

Filter search by ...

Subject area

Subject area (Metadata): *Biology*

Selected subcorpus:

Total documents: 50 (25%)

Total tokens: 211.722 (22.2%)

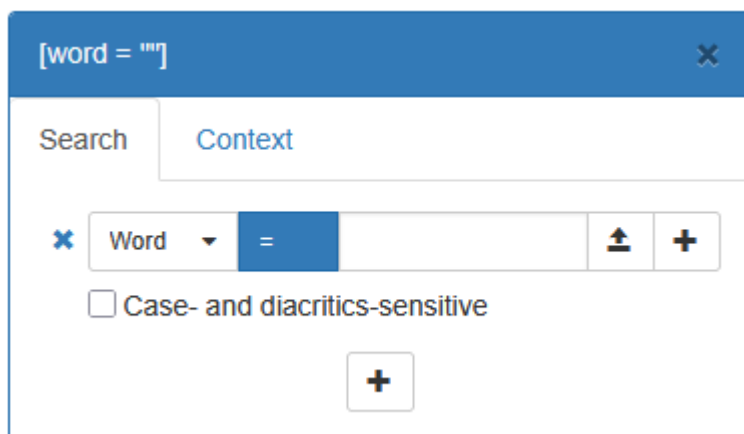
For a detailed description of the metadata, see the section [Metadata categories](#).

Advanced search

The query builder

The basic building block in the query builder is the *token box* (see below). Each box represents a token – usually just a single word – or a simple repetition of tokens; when multiple tokens are used, they are matched in order from left to right.

You can use the query builder to create complex queries without writing CQL (here: Corpus Query Language). Therefore, it is easy to use.



A token box in the querybuilder has two tabs: Search and Context.

The tab Search

The tab Search contains a set of attributes a token in the corpus must have to be matched by the query. By clicking the +-button on the right hand side of this token, you can add new attributes (see below). Then enter a value that the attribute must have for the token to be found. The search command `Lemma=lief` and `Part of Speech=Noun` for example excludes all forms of *lief* as an adjective.

The CQL query generated to match this token (the *token query*) in the corpus is displayed in the top bar of the box, to help you understand what is happening internally. The following applies to our example:

The screenshot shows a query builder window with a title bar containing the query: `[lemma = "lief" & pos_head = "n"]`. Below the title bar are two tabs: "Search" and "Context". Under the "Search" tab, there are two rows of attribute restrictions. The first row has a dropdown menu set to "Lemma", an equals sign, a text input field containing "lief", and two buttons: a plus sign and a minus sign. Below this row is a checkbox labeled "Case- and diacritics-sensitive" which is unchecked. Between the two rows is the word "AND". The second row has a dropdown menu set to "Part ...", an equals sign, a dropdown menu set to "Noun", and two buttons: a plus sign and a minus sign. Below the second row is another plus sign button.

Token attributes

Specifying token attributes is similar to the Extended Search form. Select which attribute a token should have, and enter the value that the attribute must have for the token to be matched. Attributes in the query builder are interpreted as *regular expressions*. Note that this is different from the Extended Search, where token patterns use wildcards.

Going beyond single-attribute token queries, a token box also allows you to combine several attributes and to specify repetition options.

Adding attributes to a token box

Using the +-button, new attributes can be added. Two options exist: *AND* and *OR*.

The *AND* option creates a new attribute restriction that a token must match in addition to the ones which were already there. As an example: suppose we want to match *zijn* ('to be') as a verb, not as a pronoun. First, fill in the attribute Lemma with value *zijn*, then click +, choose *AND*, and choose the value Verb for Woordsoort.

The screenshot shows a query builder window with a title bar containing the query: `[lemma = "zijn" & pos_head = "ww"]`. Below the title bar are two tabs: "Search" and "Context". Under the "Search" tab, there are two rows of attribute restrictions. The first row has a dropdown menu set to "Lemma", an equals sign, a text input field containing "zijn", and two buttons: a plus sign and a minus sign. Below this row is a checkbox labeled "Case- and diacritics-sensitive" which is unchecked. Between the two rows is the word "AND". The second row has a dropdown menu set to "Part ...", an equals sign, a dropdown menu set to "Verb", and two buttons: a plus sign and a minus sign. Below the second row is another plus sign button.

Similarly, creating a new attribute using *OR* will create a token query matching tokens that have the original attribute *or* the new attribute. For instance, enter Word *er* first, add a new attribute with the *OR* option and enter Adverb as Part of Speech to match tokens with Part of Speech tag Adverb or with word form equal to *er*.

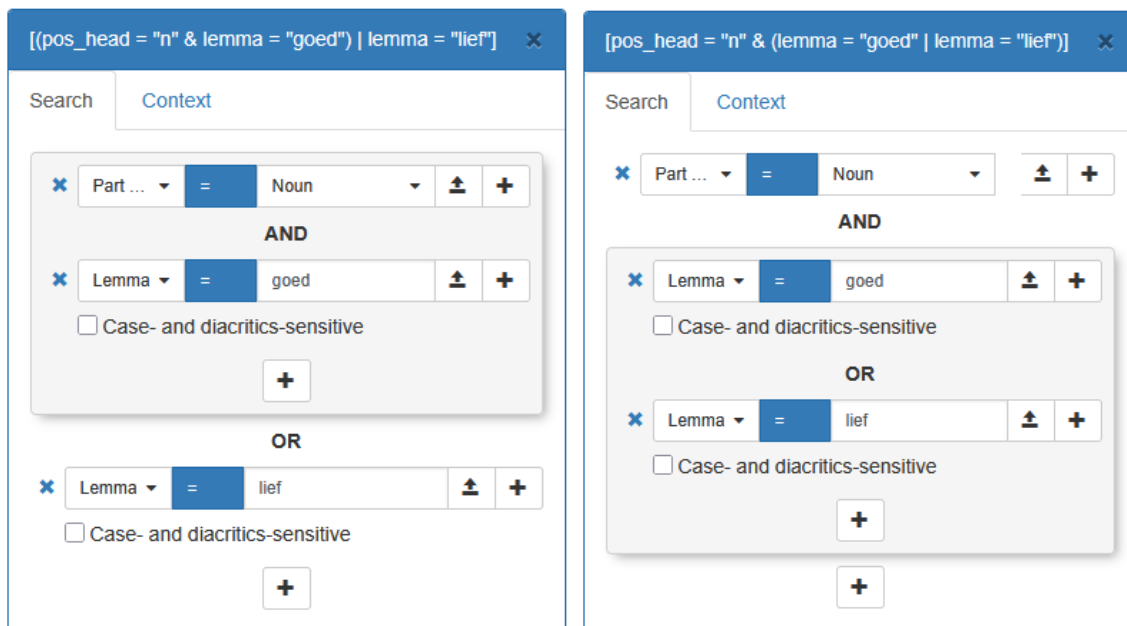
The screenshot shows a query builder window with a title bar containing the query: `[word = "er" | pos_head = "bw"]`. Below the title bar are two tabs: "Search" and "Context". Under the "Search" tab, there are two rows of attribute restrictions. The first row has a dropdown menu set to "Word", an equals sign, a text input field containing "er", and two buttons: a plus sign and a minus sign. Below this row is a checkbox labeled "Case- and diacritics-sensitive" which is unchecked. Between the two rows is the word "OR". The second row has a dropdown menu set to "Part ...", an equals sign, a dropdown menu set to "Adverb", and two buttons: a plus sign and a minus sign. Below the second row is another plus sign button.

Function of the two +-buttons in a token box

The difference between the +-sign on the right of an attribute and the one below it, is that the +-sign on the right keeps the newly added attribute 'within a subclause'. This is most easily explained by means of an example.

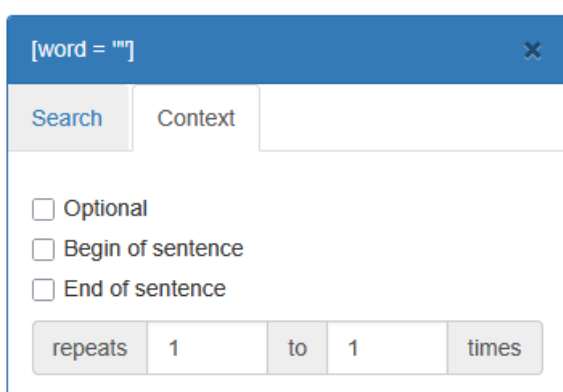
Suppose we want to search for either *goed* or *lief*, used as a noun. If we add the attributes in the order Part of Speech=Noun AND Lemma=*goed*, OR Lemma=*lief* using the +-signs **below** the attributes, as in the left screenshot below, we get the token query [(pos_head = "n" & lemma = "goed") | lemma = "lief"]. This will also match adjective forms of *lief*, so this is not what we were after.

If, on the other hand, we add OR lemma=*lief* with the +-sign to the **right** of the attribute Lemma=*goed*, it will be inserted in a subclause (Lemma=*goed* OR Lemma=*lief*), thus resulting in the correct query [pos_head = "n" & (lemma = "goed" | lemma = "lief")], as shown in the right screenshot below.



The tab Context

The tab Context specifies the contextual properties, such as whether the token occurs at the end of a sentence, and the repetition pattern:

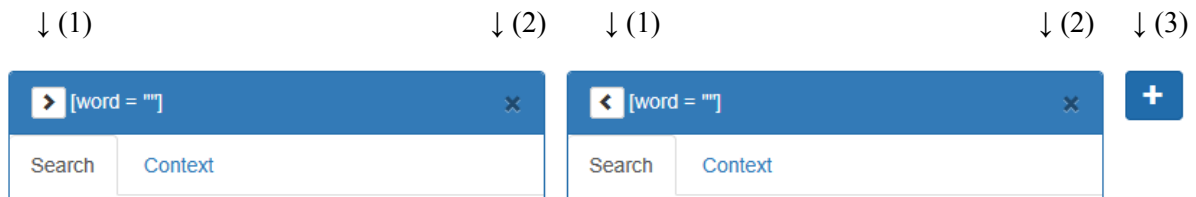


Managing sequences of token boxes

There are three ways to manage the sequence and the number of token boxes:

- *Rearrange* a token by clicking on the arrow in the top-left corner of a box (1). This arrow only appears if there are multiple token boxes.
- *Delete* a token by clicking the x in the top-right corner of a box (2).

- *Create a new token box* by clicking the + -button next to the upper right corner of the utmost right token box (3).



Uploading value lists in the query builder

It is also possible to upload a list of values, separated by a white space. To do so, click the upload button (with the arrow pointing upwards) and select a text file. Tokens will then be matched for any of the values from the file.

Note that this function only works for *.txt-files. If you are using a text editor like Word, you have to save your file as a *.txt file or you can copy and paste the values into a *.txt file first.

After uploading a file, the text can be edited by clicking the yellow marked file name in the text field. Editing the text is temporary and will not modify your original file.

To remove an uploaded file and go back to typing a value, click on the cross (x) next to the yellow text box. Another possibility to clear the uploaded values is by clicking the yellow marked text field and then pressing the Clear button on the bottom left corner of the Edit box. Using the Reset button will start a complete new search.

Within

Like in Expert Search, it is possible to apply your search query to all documents of the corpus, which is the default setting. However, it may be desirable to limit a search to a document, a sentence or an utterance. The main reason to do this is to exclude multi-word matches that stretch over a sentence boundary. To do so you can click on these respective buttons.

Copy to CQL editor

You can use the query builder to create complex queries without writing CQL. Any time a query is created in the querybuilder, it can be copied to the CQL editor, where you can further edit the query by hand. This will take you automatically to the Expert Search screen, after which you can start the search or adjust the query if desired.



Expert search

The Corpus Query Language (CQL) editor allows you to type your own CQL query, to copy your query into the query builder (in Advanced Search), to import a previously downloaded query and to upload a tab separated list of values to substitute for gap values (see below for further explanation).

CQL queries are expressions built up with the help of a few sequence operators and brackets from basic blocks enclosed by square brackets, in each of which one or more token attributes are specified.

In CQL, spaces only affect a search if they are included in quotes. Whether the search command is [word="mevrouw"] or [word = "mevrouw"] (or just "mevrouw") does not make any difference to the result. Between the queries [word="mevrouw"] and [word=" mevrouw"], however, there is a difference. The first search results in exactly 5 hits, but the second one in zero!

Some examples:

- Simple: [\[word="vrouw"\]](#), e.g. the attribute word matches the regular expression *vrouw*; [\[word!="vrouw"\]](#), e.g. the attribute word does **not** match the regular expression *vrouw*; [\[word=".*vrouw"\]](#) matches all words ending with *vrouw*, including *vrouw* itself. (Note that [\[word="*vrouw"\]](#) will not give any results, because in Expert Search an asterisk is not a wildcard but a repetition operator.)
- Combination of attributes (combining operators are &, |, !), e.g. [\[word="hoop|geloof|liefde"\]](#) matches either the word *geloof*, the word *hoop* or the word *liefde*.
- The empty [] matches any token, e.g. [\[lemma="man"\]\[\]{}3\[lemma="zijn"\]](#) matches a sequence of *man* followed by *zijn* with three arbitrary tokens in between.
- Operators |, & and parentheses () and the repetition operators (+, *, ? and {}) can be used to build complex sequence queries. Example: ["eerste" "niveau" | "laatste" "hoofdstuk"](#), matching any sequence of *laatste man* or *eerste vrouw*. Note that, while most queries up to this point could also have been constructed with the query builder, we really need the power of CQL from here on.

This short list does not cover all CQL features. For more detailed information on how to write CQL, please consult the short [Appendix: Corpus Query Language](#), which contains further pointers.

Copy to CQL editor

When the query is relatively simple – like [\[pos_head="adj"\]\[lemma="examen"\]](#) – it can also be imported into the querybuilder using the *Copy to query builder* button. This will take you automatically to the Advanced Search screen, after which you can start the search or adjust the query if desired.

A message will be displayed next to the button if the query couldn't be parsed.

Import query

If you have entered a search query, you can find it back by clicking the History button. On the right hand side you can select Download as file in the drop-down menu (default value is Search) and save the file. (For a more elaborate description of the History button see [Simple Search](#))

Previously saved queries can be used again by uploading them through the Import query button.

Gap filling

Use this button to upload a Tab Separated Values (TSV) file, which is a simple text format for storing data in a tabular structure. Each record in the table is one line of the text file. Each field value of a record is separated from the next by a tab character. It is also possible to upload a plain text file (.txt) that has the same properties.

A *.tsv file or a comparable *.txt file enables you to complete a query with marked gaps.


If, for instance, you are interested in the distribution of adjectives you can create this query in the Corpus Query Language field:

```
[lemma="@@"][pos_head="adj"][lemma="@@"]
```

By clicking Gap-filling you can upload a file with a tab-separated list of values from your computer to substitute them for the gap values, i.e. the at signs (@@) in your query. After the upload your values will appear in a separate box:

Corpus Query Language:

```
[lemma="@@"][pos_head="adj"][lemma="@@"]
```

Copy to query builder
Import query
Gap-filling 

de	man
een	vrouw
het	kind

The values in the first column - *de, een, het* - will be entered at the position of the first gap (@@) and the values in the second column - *man, vrouw, kind* - at the position of the second gap. With these values, gap-filling yields the following results:

Before Hit	Hit	After Hit	Lemma	Part of Speech (full)
Geest_S9_A_7.0 ...dapperheid toon . nooit is	een oud vrouwtje	dat dreigt overreden te worden...	een oud vrouw	lid(onbep,stan,agr) adj(prenom,basis,zonder) n(soort,ev,dim,onz,stan)
...je uh , filler weer	een oud vrouwtje	wil gaan vermoorden over de...	een oud vrouw	lid(onbep,stan,agr) adj(prenom,basis,zonder) n(soort,ev,dim,onz,stan)
Soc_S1_A_3.1 ...zowel bekijken bij leken dus	de gewone man	in de straat . wat...	de gewoon man	lid(bep,stan,rest) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
...leek . de visie van	de modale man	in de straat . wat...	de modaal man	lid(bep,stan,rest) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
Soc_S2_A_1.10 ...groot de gemiddelde vrouw en	de gemiddelde man	zijn . en vervolgens gaan...	de gemiddeld man	lid(bep,stan,rest) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
...vrouw inderdaad kleiner is dan	de gemiddelde man	. dus als je al...	de gemiddeld man	lid(bep,stan,rest) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
...verschil tussen de kleinste en	de grootste man	. Das wil wordt dan...	de groot man	lid(bep,stan,rest) adj(prenom,sup,met-e,stan) n(soort,ev,basis,zijd,stan)
Soc_S2_A_1.9 ...in de jaren zestig op	een jonge vrouw	. uh , filler die...	een jong vrouw	lid(onbep,stan,agr) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
Bio_S1_B_1.0 ...levend organisme zien dan kan	't kleinste kind	zeggen dat leeft dat	het klein kind	lid(bep,stan,evon) adj(prenom,sup,met-e,stan) n(soort,ev,basis,onz,stan)
Bio_S4_A_12.0 ...dat in het bloed van	een zwangere vrouw	dat daar kleine stukjes DNA...	een zwanger vrouw	lid(onbep,stan,agr) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
...als je bloed pakt bij	een zwangere vrouw	en je kijkt in het...	een zwanger vrouw	lid(onbep,stan,agr) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
...is van de foetus bij	een zwangere vrouw	is ongeveer tien procent	een zwanger vrouw	lid(onbep,stan,agr) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)
Bio_S4_A_6.0 ...gebruik van de eicel van	een gezonde vrouw	. goed ma * d...	een gezond vrouw	lid(onbep,stan,agr) adj(prenom,basis,met-e,stan) n(soort,ev,basis,zijd,stan)

This mimics the functionality to upload a list of values in the Extended Search and Advanced Search interfaces.

Please note that for this to work, you do need to enter @@ in the field where you want the substitution to take place. An empty field ([]) will match any term.

Viewing results

Results can be viewed in two ways: Per hit (hit is defined as one token or a group of tokens that matched the query), or Per document (each document listed contains at least one hit).

Per Hit view

Click a hit – i.e. a line with the bold word(s) in the column Hit – to display the properties and values of the hit (in the following example **uit een boek**). Click the hit again to close.

Document id: Exact_S4_A_13.1 Before Hit ▾ Hit ▾ After Hit ▾ Lemma Part of Speech (full)

Exact_S4_A_13.1

...is gebaseerd op een oefening **uit een boek** ma * d ik heb...

uit een boek vz(init) lid(onbep,stan,agr) n(soort,ev,basis,onz,stan)

...dit is een beetje het niveau dat ik op het examen zou vragen rond wrijving dit is gebaseerd op een oefening **uit een boek** ma * d ik heb het nen * d hoop complexer gemaakt . ik geef ook . in het masterjaar geef ik voertuigtechnologie en voertuigontwerp en ik heb mij daar een beetje op gebaseerd . we gaan hier nen * d auto gaan ontwerpen en vooral gaan we eens gaan...

Property	value
Word	uit een boek
Lemma	uit een boek
Part of Speech (full)	vz(init) lid(onbep,stan,agr) n(soort,ev,basis,onz,stan)

Hit rows are always preceded by a row containing the document title in which those hits occurred, in this case *Exact_S4_A_13.1*. The document titles can be toggled on or off by using the Hide Titles (or Show Titles when titles are hidden) button at the bottom of the page. If you hover the mouse over the title, the identification number of the document appears, which is identical to the document title here.

Sorting results

Click on any of the column headings to sort the hits on values within the column, clicking again inverts the sorting. Extra sorting options are given when clicking on Before hit, Hit and After hit: you can sort by Word, Lemma, Part of Speech, Part of Speech (full) and other annotations.

Before Hit ▾ Hit ▲ After Hit ▾

een oefening uit e

komt eigenlijk uit e

estiende kaart uit e

functie juist doet zor

wat die stap deed wa

stien delicaat selecte

ienten...

va

u

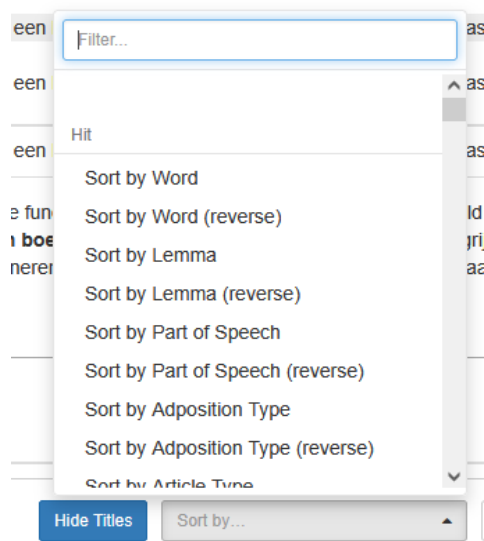
u

v

Sort by...

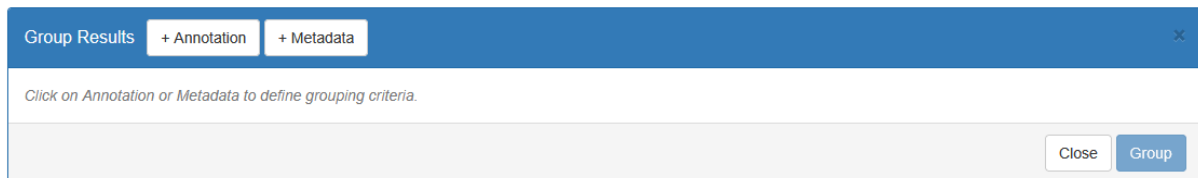
- Basics
- Word
- Lemma
- Part of Speech
- Part of Speech (full)
- More annotations
- Adposition Type
- Article Type
- Case
- Conjunction Type

You can also sort the results by means of the drop-down menu at the bottom of the page (Sort by...), which offers you the possibility to sort by various attributes for Hit, Before hit, After hit and Metadata.



Grouping results

It is possible to group the results by clicking on the button Group Results, after which the following menu appears:



Results can be grouped by Annotation and by Metadata.

By clicking +Annotation you can group by the first word, by all words or by specific words, whether within the hit, before the hit or after the hit, and based on the annotation Word, Lemma or Part of Speech, or even individual Part of Speech features (listed under more annotations). When grouping by the first word or specific words, you can also group from the end of the hit. The default grouping is grouping all words within the hit using annotation Word. Clicking +metadata allows you to group by metadata assigned to the document (Subject Area).

By clicking the Case sensitive box it is possible to distinguish between case sensitive and case insensitive.

The example below is grouped by the first word before the hit. The example dynamically updates when the grouping options are changed.

Group Results + Annotation + Metadata

first Word before hit ✕

I want to group on the first word ▾ before the hit ▾ using annotation Word ▾

Case sensitive:

je al rapper dan dan week drie zou mogen gaan bekijken
je a rapper dan dan week drie zou mogen gaan bekijken

Clear Group

Click a group to show or hide hits within that group, as shown below. Click once more on the group to close it again. If more than twenty hits are found in a document, you can make them appear by clicking on Load more concordances.

Group	#hits in group	Relative frequency (hits)
volgende	248	0.026%
vorige	140	0.0147%

[View detailed concordances](#)
[Load more concordances](#)

Before Hit	Hit	After Hit
... de heb ik vorige	week	al over verteld . dat...
... dat gezien bij de vorige	week	. Turner Klinefelter . grote...
... was eigenlijk wat we vorige	week	hebben gezien . dat bijvoorbeeld...
... wat ik hier vorige	week	vertelde . ben ook vandaag...
... daar waren we gekomen vorige	week	. begrippen . energie uiteraard...
... daar waren we gekomen vorige	week	. we hadden daar een...
... hebde * d dat vorige	week	ook gemerkt . moet ge...
... dat . dat is vorige	week	twee weken geleden . dat...
... ook nu en dan vorige	week	en volgende week mijne *...
... volia goed vorige	week	hebben we voornamelijk de indeling...
... al lang sedert vorige	week	woensdag . ja we gaan...
... 'k heb u vorige	week	gezegd . ja . in...
... puntje A van de vorige	week	. ja . dus geen...
... loopt eigenlijk tot tot vorige	week	waar ik de definitie van...
... nog eventjes wat we vorige	week	gezien hebben dus vorige week...
... week gezien hebben dus vorige	week	hebben we equivalentierelaties afgewerkt en...
... ik opfris hebben we vorige	week	een behoudswet gezien . net...
... behoudswet gaan zien . vorige	week	. een behoudswet die je...
... en die abus van vorige	week	laten voor wat ze is...
... herinner je wat ik vorige	week	ook gezegd heb 't is...

[View detailed concordances](#)
[Load more concordances](#)

de	15	0.00157%
een	14	0.00147%
per	9	0.000944%

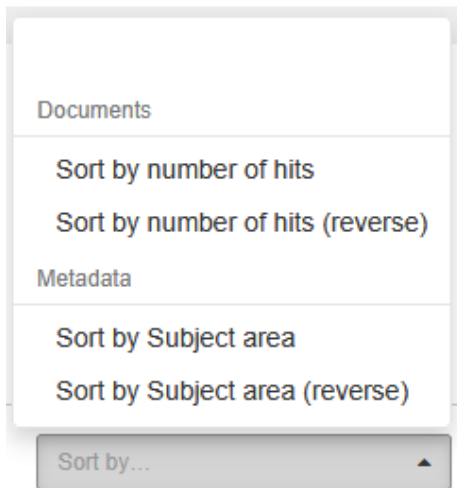
Click on View detailed concordances to go back to the normal hits view to see more detailed information for the hits in this group. The button Go back to grouped results brings you back to the list of groups.

Per Document view

Sorting results

Click on any of the column headings to sort the documents by Document (name) or Hits within that column, clicking again inverts the sorting.

You can also sort the results by means of the drop-down menu at the bottom of the page (Sort by...), which offers you the possibility to sort by Documents (e.g. the number of hits for your search query) and by Subject area.



Grouping results

Results Per Document can be grouped by metadata assigned to the document (Subject area).

Exporting results

The search results – both Per Hit as Per Document – can be exported by using the Export or the Export for Excel button at the bottom right of the page. The first button transfers the search results – including all metadata – to a Comma-Separated Values-file. These CSV-files consist only of text data, which makes it easy to implement (read and/or write) them into a spreadsheet or database program. The second button offers the possibility to export the results – including all metadata – to a CSV-file for use with Excel.

Grouped results can be exported in the same way. However, if you would like to have the metadata with each concordance of a group, you must first click on the blue bar of a specific group and then on View detailed concordances. The results you then see can be exported by the use of the Export buttons. This operation must be carried out for each individual group you wish to export.

Information about a document

Click on a document title or the chain icon in the per hit view to open this document in a new window: the Content window.

Content

Hits from the current query will be highlighted in bold in the opened document. In the case of several hits only the current hit will also appear in shadow (such as *definitie* in the example below). You can navigate from one hit to another by using the arrows at the Hits button (this button can be dragged around), and you can also browse through the pages if the document consists of more than one page.

When you hover with your mouse over a specific word in the document a pop-up will appear with the modern lemma and the option Show details. By clicking this link you will see extra information on word level.

id	Exact_S7_A_13.0.eaf.ManueleSegmentatieASRAnnotated1.clean.punct.p.1.s.80.w.8
lemma	definitie
pos	N(soort,ev,basis,zijd,stan)

en we beginnen toch nog met een **definitie definitie** van orthonale verzameling en een

Metadata

In the metadata tab, all metadata properties of the document are displayed. They provide information about Subject area and also Document Length.

Statistics

The Statistics tab shows several document statistics: the number of Tokens, Types (unique word forms), Lemmas, Part of Speech and the Type/Token ratio. It is possible to print or to download these statistics via the menu symbol right of the title Vocabulary Growth.

Exploring the corpus

The Explore tab has three subdivisions: Documents, N-grams and Statistics.

Documents

This subtab allows you to investigate the documents. It consists of two drop-down menus to specify the grouping of the metadata and to specify the way the groups are to be shown.

Pressing Search after opening the Documents subtype shows the four subject areas with information about each area.

Group	#docs in group	#tokens in group			Average document length
Humanities	50	236.201	25%	24.8%	4.725
Social sciences	50	280.184	25%	29.4%	5.604
Exact sciences	50	225.609	25%	23.7%	4.513
Biology	50	211.722	25%	22.2%	4.235

N-grams

An *N-gram* is a sequence of *N* items. This option will list the frequency of different N-grams in a (sub-)corpus.

Options

- *N-gram size*: the length of the sequence (a number from 1 to 5; default setting is 5).
- *N-gram type*: the attribute to search for. You can choose: Word (i.e. word form), Lemma and Part of Speech. If you do not specify the search term a series of arbitrary words equal to the n-gram size will be searched for.
- It is also possible to restrict to, for instance, n-grams with some slots already specified, as is shown in the following example.
- By using the Filter search by ... you can create a subcorpus for specific metadata.

Example

Within all the documents, you will find 13 occurrences of this so-called 5-gram.

Group	#hits in group	Relative frequency (hits)
de hele dag geslapen .	2	0.00021%
de twee dagen . en	1	0.000105%
de ene dag op de	1	0.000105%
de komende dagen hoop ik	1	0.000105%
de volgende dagen ter beschikking	1	0.000105%
de komende dagen op Toledo	1	0.000105%
de Internationale Dag van de	1	0.000105%
de laatste dag dit semester	1	0.000105%
de zoveel dagen en weken	1	0.000105%
De Zevende Dag dat gaan	1	0.000105%
de andere dagen on campus	1	0.000105%
de komende dagen even bekijken	1	0.000105%

Statistics (frequency lists)

Here, you can produce frequency lists for the corpus. It is rather similar to the previous option, but restricted to 1-grams.

Options

- *Frequency list type*: choose for Word (i.e. word forms), Lemma, Part of Speech, Part of Speech (full) or any of the other annotations.
- By using the Filter search by... you can create a subcorpus within SABeD for specific metadata.

Example

It is possible to determine the use of the most frequently used lemmata for the subject area Humanities by searching for Frequency list type Lemma and by filtering search by Subject area: *Humanities*. This results in:

Results for: Lemma frequency within documents where Subject area: Humanities

Per HitPer Document

Hits / Grouped by Lemma within hit


Total hits: 236.201 (100%)
Total groups: 9.117
Search time: 0.1s

Group Results+ Annotation+ Metadata✕

Lemma within hit ✕

I want to group on all words ▾ within the hit ▾ using annotation Lemma ▾

Case sensitive:

 **dit filmpje bespreek ik hoofdstuk**
di. film bespreken il hoofdstuk

Clear Group

« 1 2 3 4 6 11 » table hits

Group	#hits in group	Relative frequency (hits)
.	18.214	7.71%
de	8.651	3.66%
een	8.377	3.55%
zijn	8.210	3.48%
dat	7.643	3.24%
die	5.950	2.52%
en	5.727	2.42%
het	5.658	2.4%
van	4.930	2.09%
in	4.240	1.8%
je	4.084	1.73%
we	3.074	1.3%
hebben	3.028	1.28%
dus	2.315	0.98%
dan	2.288	0.969%
als	2.151	0.911%

Appendix: Corpus Query Language

BlackLab supports Corpus Query Language, a full-featured query language introduced by the IMS Corpus WorkBench (CWB) and also supported by the Lexicom Sketch Engine. It is a standard and powerful way of searching corpus.

The basics of Corpus Query Language is the same in all three projects, but there are a few minor differences in some of the more advanced features, as well as some features that are exclusive to some projects. For most queries however, this will not be an issue.

This page will introduce the query language and show all features that BlackLab supports. If you want to learn even more about CQL, see [CWB COP Query Language Tutorial](#) and [Sketch Engine Corpus Query Language](#).

CQL support

For those who already know CQL, here's a quick overview of the extent of BlackLab's support for this query language. If there is a feature we don't support, yet is important to you, please let us know. If it's quick to add, we may be able to help you out.

Supported features

BlackLab currently supports (arguably) most of the important features of Corpus Query Language:

- Matching on token annotations (also called properties or attributes), using regular expressions and =, !=, !. Example: [word="bank"] (or just "bank")
- Case/accnt-sensitive matching. Note that, unlike in CWB, case-INsensitive matching is currently the default. To explicitly match case/accnt-insensitivity, use "(?i)...". Example: "(?i)Mr\." "(?i)Banks"
- Combining criteria using &, | and !. Parentheses can also be used for grouping. Example: [lemma="bank" & pos="V"]
- Match-all pattern [] matches any token. Example: "a" [] "day"
- Regular expression operators +, *, ?, {n}, {n,m} at the token level. Example: [pos="AA"]+
- Sequences of token constraints. Example: [pos="AA"] "cow"
- Operators |, & and parentheses can be used to build complex sequence queries. Example: "happy" "dog" | "sad" cat"
- Querying with tag positions using e.g. <s> (start of sentence), </s> (end of sentence), <s/> (whole sentence) or <s> ... </s> (equivalent to <s/> containing ...). Example: <s> "The" . XML attribute values may be used as well, e.g. <ne type="PERS"/> ("named entities that are persons").
- Using within and containing operators to find hits inside another set of hits. Example: "you" "are" within <s/>
- Using an anchor to capture a token position. Example: "big" A:[]. Captured matches can be used in global constraints (see next item) or processed separately later (using the Java interface; capture information is not yet returned by BlackLab Server). Note that BlackLab can actually capture entire groups of tokens as well, similarly to regular expression engines.
- Global constraints on captured tokens, such as requiring them to contain the same word. Example: "big" A:[] "or" "small" B:[] :: A.word = B.word

See below for features not in this list that may be added soon, and let us know if you want a particular

feature to be added.

Differences from CWB

BlackLab's CQL syntax and behaviour differs in a few small ways from CWBs. In future, we'll aim towards greater compliance with CWB's de-facto standard (with some extra features and conveniences).

For now, here's what you should know:

- Case-insensitive search is currently the default in BlackLab, although you can change this if you wish. CWB and Sketch Engine use case-sensitive search as the default. We may change our default in a future major version.
If you want to switch case-/diacritics-sensitivity, use "(?-i).." (case-sensitive) or "(?i).." (case-insensitive, usually the default). CWBs %cd flags for setting case/diacritics-sensitivity are not (yet) supported, but will be added.
- If you want to match a string literally, not as a regular expression, use backslash escaping: "e\.g\.". %l for literal matching is not yet supported, but will be added.
- BlackLab supports result set manipulation such as: sorting (including on specific context words), grouping/frequency distribution, subsets, sampling, setting context size, etc. However, these are supported through the REST and Java APIs, not through a command interface like in CWB. See [BlackLab Server overview](#).
- Querying XML elements and attributes looks natural in BlackLab: <s/> means "sentences", <s> means "starts of sentences", <s type='A'> means "sentence tags with a type attribute with value A". This natural syntax differs from CWBs in some places, however, particularly when matching XML attributes. While we believe our syntax is the superior one, we may add support for the CWB syntax as an alternative.
We only support literal matching of XML attributes at the moment, but this will be expanded to full regex matching.
- In global constraints (expressions occurring after ::), only literal matching (no regex matching) is currently supported. Regex matching will be added soon. For now, instead of A:[] "dog" :: A.word = "happy|sad", use "happy|sad" "dog".
- To expand your query to return whole sentences, use <s/> containing (...). We don't yet support CWBs expand to, expand left to, etc., but may add this in the future.
- The implication operator -> is currently only supported in global constraints (expressions after the :: operator), not in regular token constraints. We may add this if there's demand for it.
- We don't support the @ anchor and corresponding target label; use a named anchor instead. If someone makes a good case for it, we will consider adding this feature.
- backreferences to anchors only work in global constraints, so this doesn't work: A:[] [] [word = A.word]. Instead, use something like: A:[] [] B:[] :: A.word = B.word. We hope to add support for these in the near future, but our matching approach may not allow full support for this in all cases.

(Currently) unsupported features

The following features are not (yet) supported:

- intersection, union and difference operators. These three operators will be added in the future. For now, the first two can be achieved using & and | at the sequence level, e.g. "double" [] & []

"trouble" to match the intersection of these queries, i.e. "double trouble" and "happy" "dog" | "sad" "cat" to match the union of "happy dog" and "sad cat".

- `_` meaning "the current token" in token constraints. We will add this soon.
- `lbound`, `rbound` functions to get the edge of a region. We will probably add these.
- `distance`, `distabs` functions and `match`, `matchend` anchor points (sometimes used in global constraints). We will see about adding these.
- using an XML element name to mean 'token is contained within', like `[(pos = "N") & !np]` meaning "noun NOT inside in an tag". We will see about adding these.
- a number of less well-known features. If people ask, we will consider adding them.

Using Corpus Query Language

Matching tokens

Corpus Query Language is a way to specify a "pattern" of tokens (i.e. words) you're looking for. A simple pattern is this one:

```
[word="man"]
```

This simply searches for all occurrences of the word "man". If your corpus includes the per-word properties `lemma` (i.e. headword) and `pos` (part-of-speech, i.e. noun, verb, etc.), you can query those as well. For example, to find a form of word "search" used as a noun, use this query:

```
[lemma="search" & pos="NOU-C"]
```

This query would match "search" and "searches" where used as a noun. (Of course, your data may contain slightly different part-of-speech tags.)

The first query could be written even simpler without brackets, because "word" is the default property:

```
"man"
```

You can use the "does not equal" operator (`!=`) to search for all words except nouns:

```
[pos != "NOU-C"]
```

The strings between quotes can also contain wildcards, of sorts. To be precise, they are [regular expressions](#), which provide a flexible way of matching strings of text. For example, to find "man" or "woman", use:

```
"(wo)?man"
```

And to find lemmata starting with "under", use:

```
[lemma="under.*"]
```

Explaining regular expression syntax is beyond the scope of this document, but for a complete overview, see [here](#).

Sequences

Corpus Query Language allows you to search for sequences of words as well (i.e. phrase searches, but with many more possibilities). To search for the phrase "the tall man", use this query:

```
"the" "tall" "man"
```

It might seem a bit clunky to separately quote each word, but this allows us the flexibility to specify exactly what kinds of words we're looking for. For example, if you want to know all single adjectives used with man (not just "tall"), use this:

```
"an? | the" [pos="AA"] "man"
```

This would also match "a wise man", "an important man", "the foolish man", etc.

Regular expression operators on tokens

Corpus Query Language really starts to shine when you use the regular expression operators on whole tokens as well. If we want to see not just single adjectives applied to "man", but multiple as well:

```
"an? | the" [pos="AA"]+ "man"
```

This query matches "a little green man", for example. The plus sign after [pos="AA"] says that the preceding part should occur one or more times (similarly, * means "zero or more times", and ? means "zero or one time").

If you only want matches with two or three adjectives, you can specify that too:

```
"an? | the" [pos="AA"] {2,3} "man"
```

Or, for two or more adjectives:

```
"an? | the" [pos="AA"] {2,} "man"
```

You can group sequences of tokens with parentheses and apply operators to the whole group as well.

To search for a sequence of nouns, each optionally preceded by an article:

```
("an? | the"? [pos="NOU-C"])+
```

This would, for example, match the well-known palindrome "a man, a plan, a canal: Panama!"

Punctuation

In BlackLab, punctuation tends to not be indexed as a separate token, but as a property of a word token - CWB and Sketch Engine on the other hand tend to index punctuation as a separate token instead. You certainly could choose to index punctuation as a separate token in BlackLab, by the way -- it's just not commonly done. Both approaches have their advantages and disadvantages, and of course the choice affects how you write your queries.

It is possible to search for punctuation marks. E.g. to find occurrences of the word "want" preceded by a comma use the following query:

```
[punctBefore="," & word="want"]
```

To find occurrences of the lemma "krant" that are followed by an exclamation mark, use:

```
[lemma="krant" & punctAfter="!"]
```

Some punctuation marks have a special function in regular expressions and therefore must be preceded by a backslash (\) when used in queries. For instance, to search for a period (.) after the word **"geweest"**, use:

```
[word="sentence" & punctAfter="\." ]
```

Case- and diacritics-sensitivity

CWB and Sketch Engine both default to (case- and diacritics-)sensitive search. That is, they exactly match upper- and lowercase letters in your query, plus any accented letters in the query as well. BlackLab, on the contrary, defaults to *IN*sensitive search (although this default can be changed if you like). To match a pattern sensitively, prefix it with "(?-i)":

```
"(?-i) Panama"
```

If you've changed the default search to sensitive, but you wish to match a pattern in your query insensitively, prefix it with "(?i)":

```
[pos="( ?i) NOU-C"]
```

Although BlackLab is capable of setting case- and diacritics-sensitivity separately, it is not yet possible from Corpus Query Language. We may add this capability if requested.

Matching XML elements

Corpus Query Language allows you to find text in relation to XML elements that occur in it. For example, if your data contains sentence tags, you could look for sentences starting with "the":

```
<s>"the"
```

Similarly, to find sentences ending in "that", you would use:

```
"that"</s>
```

You can also search for words occurring inside a specific element. Say you've run named entity recognition on your data and all person names are surrounded with <person>...</person> tags. To find the word "baker" as part of a person's name, use:

```
"baker" within <person/>
```

Note the forward slash at the end of the tag. This way of referring to the element means "the whole element". Compare this to <person>, which means "the element's open tag", and </person>, which means "the element's close tag".

The above query will just match the word "baker" as part of a person's name. But you're likely more interested in the entire name that contains the word "baker". So, to find those full names, use:

```
<person/> containing "baker"
```

Or, if you simply want to find all persons, use:

```
<person/>
```

As you can see, the XML element reference is just another query that yields a number of matches. So as you might have guessed, you can use "within" and "containing" with any other query as well. For example:

```
( [pos="AA"]+ containing "tall") "man"
```

will find adjectives applied to man, where one of those adjectives is "tall".

Labeling tokens, capturing groups

Just like in regular expressions, it is possible to "capture" part of the match for your query in a "group".

CWB and Sketch Engine offer similar functionality, but instead of capturing part of the query, they label a single token. BlackLab's functionality is very similar but can capture a number of tokens as well. For example:

```
"an?|the" Adjectives: [pos="AA"]+ "man"
```

This will capture the adjectives found for each match in a captured group named "Adjectives".

BlackLab also supports numbered groups:

```
"an?|the" 1: [pos="AA"]+ "man"
```

Global constraints

If you tag certain tokens with labels, you can also apply "global constraints" on these tokens. This is a way of relating different tokens to one another, for example requiring that they correspond to the same word:

```
A: [] "by" B: [] :: A.word = B.word
```

This would match "day by day", "step by step", etc.