Brieven als Buit application manual

Table of contents

Introduction

This manual describes the corpus exploitation environment for the *Brieven als Buit* ('Letters as loot') corpus. The corpus application is developed by the INT. The backend of the application is the BlackLab Lucene based search engine developed for corpora with token-based annotation (http://inl.github.io/BlackLab/). The web-based frontend is a further development of the corpus-frontend application developed by INT (https://github.com/INL/corpus-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 *Brieven als Buit* corpus comprises 1033 letters and was compiled from the collection of approximately 40,000 Dutch letters (The National Archives, Kew UK) within the *Brieven als Buit / Letters as Loot* research programme, directed by prof. dr. Marijke van der Wal (Leiden University) and funded by the Netherlands Organisation for Scientific Research (NWO). The letters were sent home by sailors and others from abroad but also vice versa by those staying behind who wanted to keep in touch with their loved ones. Many letters did not reach their destinations: they were taken as loot by privateers and confiscated by the High Court of Admiralty during the naval wars fought between The Netherlands and England from the second half of the 17th century to the early 19th century. These confiscated letters of men, women and even children represent priceless material for historical linguists. They allow us to gain access to the as yet mainly unknown everyday Dutch of the past, the colloquial Dutch of people from the middle and lower classes.

All texts were tokenised, tagged with a part of speech and lemmatised. The named entities were also labelled. The linguistic annotation was done automatically, and verified manually for the entire corpus. A first online accessible version of the corpus was launched on 5 September 2013. It was followed by a second, adapted release on 18 June 2015.

This third version - released on 29 January2021 - has not only been given a new layout, but several corrections have also been made and some changes have been made to the part of speech tagset (see below).

Lemmatization

The word forms in this corpus all have a modern Dutch lemma. For words no longer used in modern Dutch, a modern lemma has been constructed using the same linguistic principles applicable to still existing words.

More information about the used lemmatization principles can be found in Marijke Mooijaart, Het lemma in the GiGaNT lexicon.

Part of speech tagging

In the context of the CLARIAH+ project, a tagset and tagging principles for the annotation of diachronic corpora of historical Dutch has been developed. This annotation layer has been added to the corpus, and can also be used to search the online corpus.

A detailed description can be found here.

The tags that were originally used for *Brieven als Buit* have been mapped to the TDN tags, which can be found in the detailed description. The most important differences are:

- the use of a type feature for different kinds of proper names instead of using different tags, such as PER, NEPER, NELOC and NEORG;
- the use of a type feature for the residual categories tagged by RES, FOREIGN and UNRESOLVED;
- the ADJ tag is replaced by AA and ART is now a subtype of PD
- multiple tags that denote the same thing have been uniformized to a single tag, such as NOU-C for NOU-C, NOU and NOU-EN and VRB for VRB and VRN.

Metadata categories

The *Brieven als Buit* corpus has been enriched with an elaborate set of metadata categories. These metadata will all be described below. In the corpus application it is possible to limit a search by filtering on metadata categories.

Letter

A letter from the *Brieven als Buit* corpus.

Year

The year(s) in which the letter(s) was (were) written, as evidenced by the date of the letter.

Text type

The type of text to which the letter belongs (business, private). Most of the letters in the corpus consist of private letters. Letters to friends and family containing both personal and business information are considered private letters.

Autograph

In the case of autographs (autograph), it has been established that the sender actually wrote the letter. In the case of non-autographs (non-autograph) the sender did not write the letter himself, but had it done by someone else. In uncertain cases it could not be determined whether the letter is an autograph or not.

Signature

The signature of the box containing letters in the archives of the High Court of Admiralty (HCA) in the National Archives in Kew, United Kingdom. A signature always starts with the letters HCA followed by a series of numbers, for example HCA 30-223.

Sender

The person who sent the letter. Please note that this is not always the actual scribe of the letter (see under Autograph).

Name

The name of the sender.

Gender

The gender of the sender (female, male, unknown).

Class

Four social layers are distinguished, based on the stratification that is common among historians (see Willem Frijhoff & Marijke Spies, 1650. *Bevochten eendracht*. Den Haag: Sdu, 1999, pp. 188-190). The four social segments of the application are: low class, middle-low class, middle-high class and high class.

The low class includes, for example, seafarers from the lowest ranks, servants, soldiers and the poor (78 documents). The middle-low class includes small shopkeepers, small farmers, seafarers from lower ranks and craftsmen (216 documents). The middle-high class consists of, for example, small businessmen, wealthy farmers, master craftsmen, captains and lower ranking officers such as mates (414 documents). The high class includes wealthy merchants, ship owners, academics, senior officials and senior officers in the army and navy (175 documents). It should be noted that high class does not refer to the highest social class of nobility and non-noble ruling classes. That upper class does not occur in the corpus. There are 150 documents of which it is not known to which class the sender belonged.

Age

Three age groups are distinguished, namely <30 (under 30), 30-50 (30 to 50 years) and >50 (over 50).

Region of residence

This designation refers to the region where a sender grew up or where he or she spent most of his or her life. The region of residence is usually a Dutch province such as Zeeland or Zuid-Holland. The corpus contains many letters from the Caribbean, sent by people who temporarily or for a longer period of time stayed in the Caribbean region, but who originally came from Zeeland or Zuid-Holland. The region of residence is then Zeeland or Zuid-Holland and not, for example, Curaçao. This characteristic is important for linguistic research into dialect differences.

Note that for Noord-Holland, a distinction is made between Noord-Holland, Amsterdam - a city that was a metropolis at the time - and Noord-Holland (excluding Amsterdam).

Relationship to addressee

The relationship, personal and/or professional, that the sender has with the addressee, such as acquaintance, employer, friend, grandson, mother, nephew/cousin or sister.

Addressee

The person to whom the letter was sent.

Name

The name of the consignee.

Place

The place to which a letter was sent, e.g. Enkhuizen or Middelburg.

Country

The country to which a letter has been sent. Note that contemporary names are used, for example Sri Lanka (and not Ceylon) and Saint Kitts (and not St. Christopher).

Region

Within the Dutch language area, the term region refers to a province or dialect area, such as Noord-Brabant, West-Vlaanderen and Zeeland. Outside the Dutch language area, it indicates a geographical region: Azië (Asia), Caraïbisch gebied (Caribbean), Noord-Europa (Northern Europe), West-Afrika (West Africa) and Zuid-Europa (Southern Europe).

Ship

The ship to which a letter was sent, e.g. the *Spiegel*.

Sent from

The four place designations that are distinguished in this tab (Place, Country, Region, Ship) have already been described above at Addressee. Please note that the names of places, countries, regions and ships are different from those found at Addressee.

Application user manual

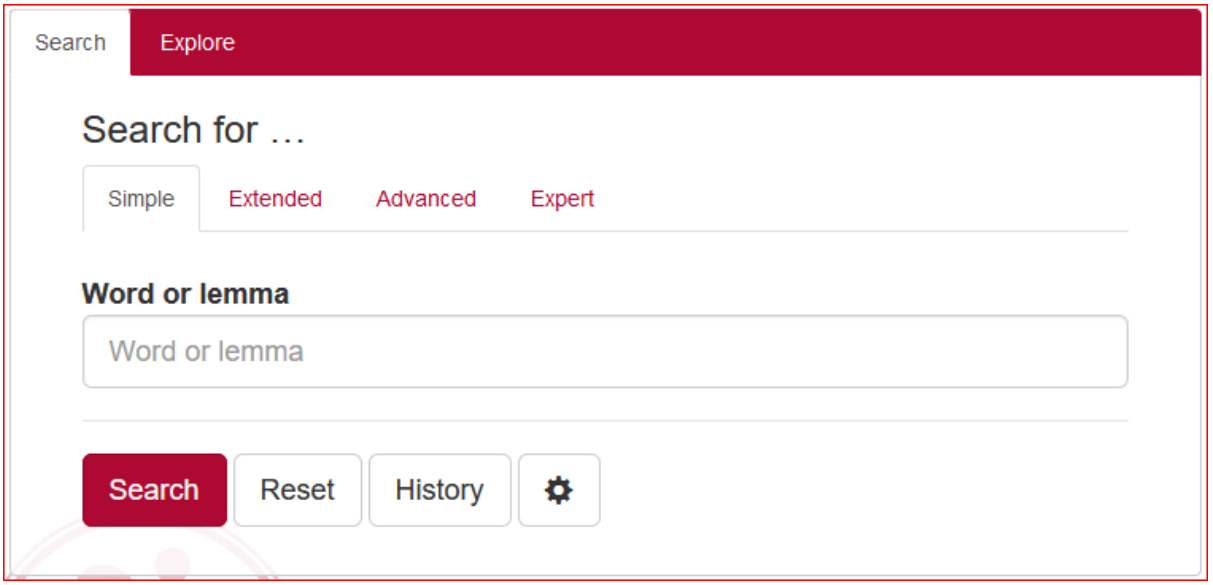
Getting started

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 the attribute Word in Extended Search:
    - Simple Search for *scepe*
    - Extended Search for Word *dochter*
- To search by lemma form (i.e. the canonical form or citation form of a set of forms (headform)), you can use Simple Search or else the attribute Lemma in Extended Search
    - Simple Search for *ziekte*
    - Extended Search for Lemma *ziekte*
- To search for words satisfying a certain pattern, use *wildcards* in Simple Search or Extended Search, or *regular expressions* in Advanced Search or Expert Search
    - words ánd lemmata starting with *ver* and ending with *len* in Simple Search
    - only word forms starting with *ver* and ending with *len* in Extended Search
    - only lemmata starting with *ver* and ending with *len* in Extended Search
    - lemmata starting with *ver*, ending in *eren* with one syllable in between in Expert Search
- To search for a multi-word pattern, e.g. all adjectives appearing before a given lemma as a noun, use the query builder in Advanced Search or use Expert Search:
    - adjectives before the lemma *huis* in query builder in Advanced Search
    - adjectives before the lemma *huis* in Expert Search
- To see which unique forms occur as a result of your search, use the Group hits by feature.
    - example Group by Lemma: different adjectives before the lemma *huis*
    - example Group by Lemma before: different words preceding the word *god*
- To explore the distribution of document properties in the corpus, use the Explore feature
    - example: characteristics about the signature
    - example: speaker age distribution

Searching the corpus

Simple search



Search

The Simple Search allows you to quickly search for specific words (e.g. *scepen*) or lemmata (e.g. *schip*). It is also possible to enter a phrase: *de 4 scepe* (words) or *het schip zijn* (lemmata). To start the search simply press enter of press the Search button.

The search field Word or lemma is provided with a list, which contains suggestions for possible search terms in alphabetical order, based on the characters typed in. Note that this only works with a single word, like *schip*. If you want to use the autocomplete option for a multi-word lemma (e.g. *Kaap de Goede Hoop*), the search query must be preceded by a quotation mark (e.g. "Kaap de Goede Hoop"). Otherwise only the last word you typed in will be automatically completed.

Keep in mind that when a historical word form corresponds with a modern Dutch lemma, you will not only find the desired historical word form, but also all word forms that can be traced back to that homonymous lemma. For instance, the search term *man* does not only result in all occurrences of *man*, but also in word forms such as *manne*, *man*, *mans*, which after all also belong to the lemma *man*. In order to only find the word form *man,* use the attribute Word in Extended Search (see over there).

Note that in Simple Search the patterns will be matched case-insensitively**:** *schip* will deliver the same results as *schip* or *Schip*. See the paragraph Grouping results in Per Hit view to see how it is nevertheless possible to distinguish between uppercase and lowercase letters.

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, *a*n* in Word or lemma matches all word forms and lemmata that start with an *a* and end with a *n*, e.g. ***aen*** (Word and also part of the Lemma ***a****anlope****n***), ***a****llee****n*** (both Word and Lemma), ***aen*** (Word and the corresponding Lemma ***aan***) but also *of* (because of the multi-word Lemma ***a****fbreke****n***).

? The question mark matches a single character once. Therefore, searching for *a?n* in Word or lemma matches *only* three-letter word forms or lemmata with an *a* and ending with a *n*, e.g. ***aen***, *an* (Lemma ***aan***), *aenden* (Lemma ***aan***+*de*).

This wildcard can be used more than once. Thus *a???n* in Word or lemma matches ***a****llee****n*** (lemma al), *aerjaen* (lemma **A**rja**n**) and ***a****llee****n*** (Lemma *alleen*).

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 Word or lemma.

History

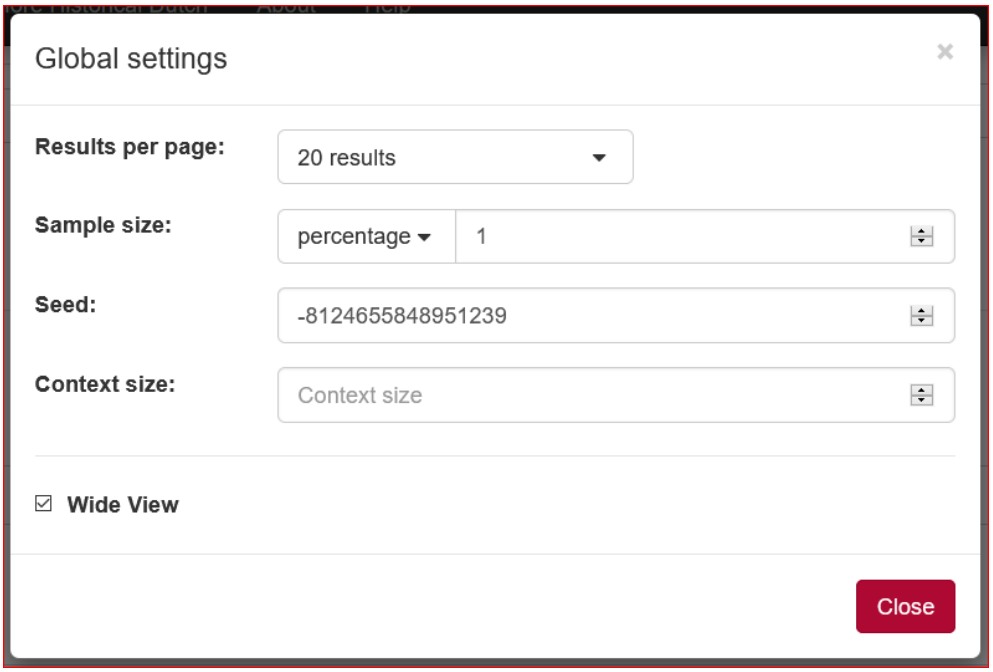
The History button will display your query history. Per search query there are several possibilities (as shown in the screenshot below): you can perform the search query again (Search), you can copy the search query as a link (Copy as link), you can download the search query as a file (Download as file),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

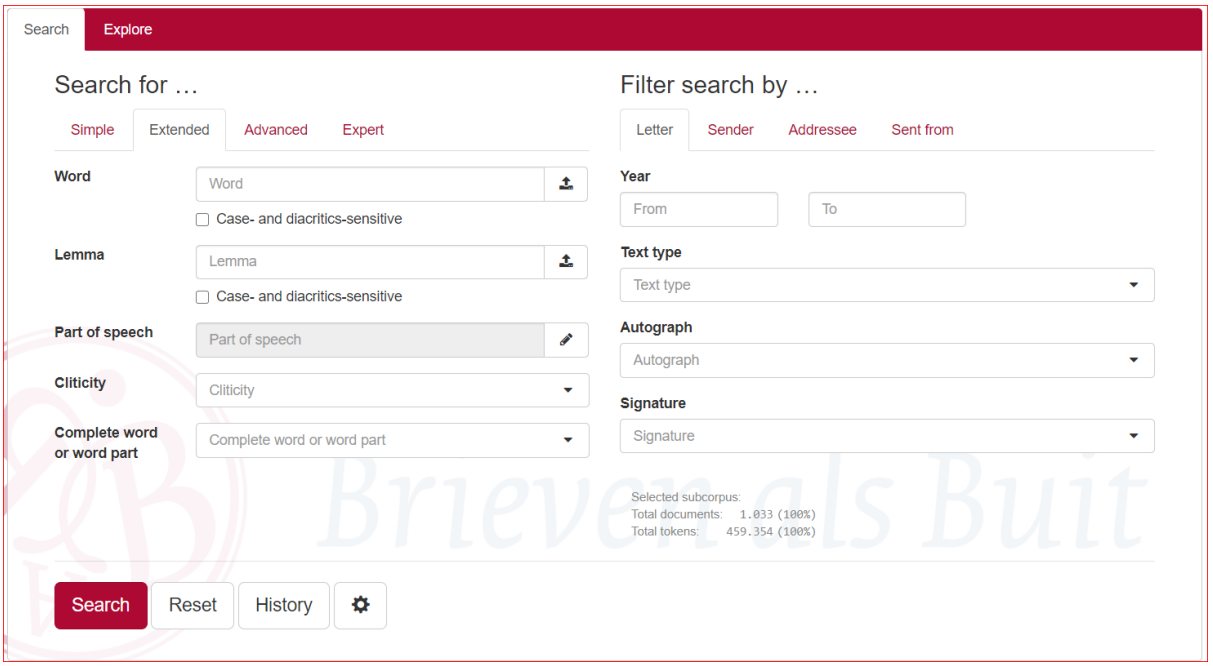
- *Results per page*: you can choose whether you want 20, 50, 100 or 200 results to be shown;
- *Sample size:* selecting a value here will instruct the search engine to return a random sample drawn from the complete result set. The sample size can be limited by
    - a percentage of the total number 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 words Before hit and After hit;
- *Wide View:* the default setting is 'small view'; you can change to Wide View by ticking the checkbox.



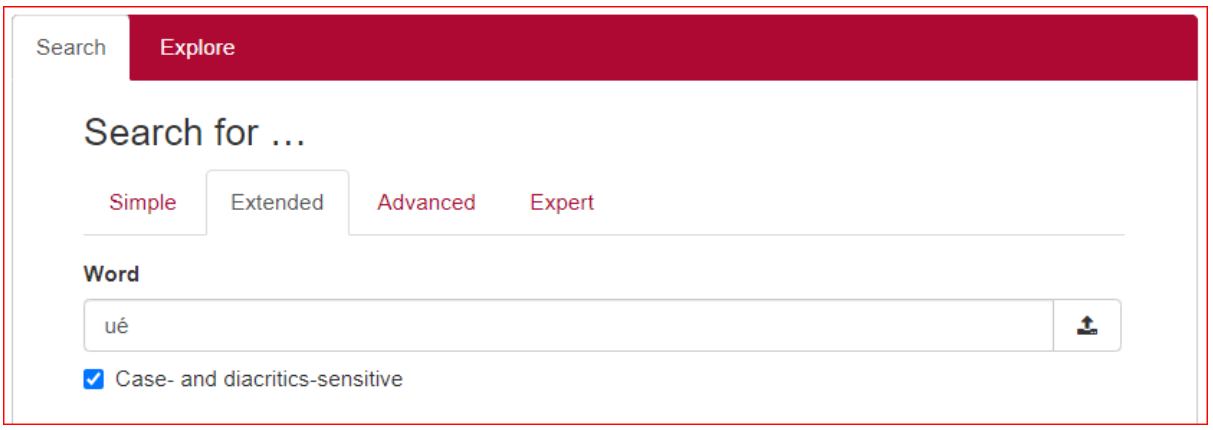
Extended search

The Extended Search allows you to find all occurrences of a *token* with its specific *attributes*. A *token* - usually just a single word - is the smallest unit within a corpus, whereas *attributes* are the different values that together make up a token.

In this corpus the five attributes you can search for are Word (more precise: word form), Lemma, Part of speech, Cliticity and Complete word or word part. All supported attributes are shown in the search form:



In the search fields Word and Lemma enter the value of the attributes (or Upload a list of values; see below) you are looking for. In the search fields Part of speech, Cliticity and Complete word or word part you can select the desired values. Then press enter or click the Search button below to execute the search and view the results. Note that the default setting for Word and Lemma in Extended search is case- and diacritics-insensitive. For example, searching for the Word *ué* (i.e. *Uwe Edelheid*) will result in all spelling variants as *UE, ue, uE, Ue, ué, Ué* and *UÉ*. In order to directly find only occurrences of the Word (form) *ué*, tick the box Case- and diacritics-sensitive under the search field Word (as shown below).



Please note that there is an important difference between the search fields Word and Lemma. As an example: entering the value *schip* in Word will only provide you with occurrences of that exact string of characters. When you enter *schip* in the search field Lemma you will - besides the lemma *schip* - also find all word forms that are linked to that lemma, such as the plurals *schepen* and *scheepen* and the spelling variants *schijp* and *schijep*.

Wildcards

In Extended 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, *a*n* in Word matches all word forms that start with an *a* and end with a *n*, e.g. word forms ***a*****e****n***, ***a*****llee****n***, ***a*****n**, ***a*****a****n*** and ***a*****ltesame****n***. Note that the same query in Lemma will give other results.

? The question mark matches a single character once. Therefore, searching for *a?n* in Lemma matches *only* three-letter values starting with an *a* and ending with a *n*, e.g. ***a*****a****n*** (and clitic word forms containing the word part *aan*).

This wildcard can be used more than once. Searching for *a???n* in Word matches the word forms *allen*, *anden*, *Armen*, *anten, alden*, whereas searching for *a???n* in Lemma matches for instance the lemmata *Anten* (word *anten*), *afijn* (word *afin*, *affin*), *Arjen* (word *Aarjen*), *Alien* (word *aelijen*), *ajuin* (word *hijun*).

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.)

In the search fields Word and Lemma it is possible to search for different values simultaneously by separating them without spaces by a vertical line, e.g. *god|man|lief* or - with the use of wildcards - *god|aan*|hond*.

For the search field Word it is also possible to search for a series of tokens by entering multiple values - including wildcards - separated by a space, e.g. *lieve vrouw, lieve *, lieve * man* or *lieve man*. Note that searching for *lief vrouw*, *lief **, *lief * man* and *lief man* in the search field Lemma will give different results!

Values at the same position in different fields are grouped together as a single token, meaning that all values in the first position of each field are grouped to match a single token.

- A single-token example: searching for the Word(form) *loop* together with the part of speech NOU-C will result in a list of all nouns containing the word form *loop*. The syntax of your query is shown in the results: [word="loop"&pos="nou\-c"].
- A multi-token example: searching for *mijn draet* in the Word(form) field and *ik dragen* in the Lemma field finds those occurrences of the bigram in which the first word is the declined form of the personal pronoun *ik* and the second belongs to the paradigm of the verb *dragen*: [word="mijn"&lemma="ik"][word="draet"&lemma="dragen"].

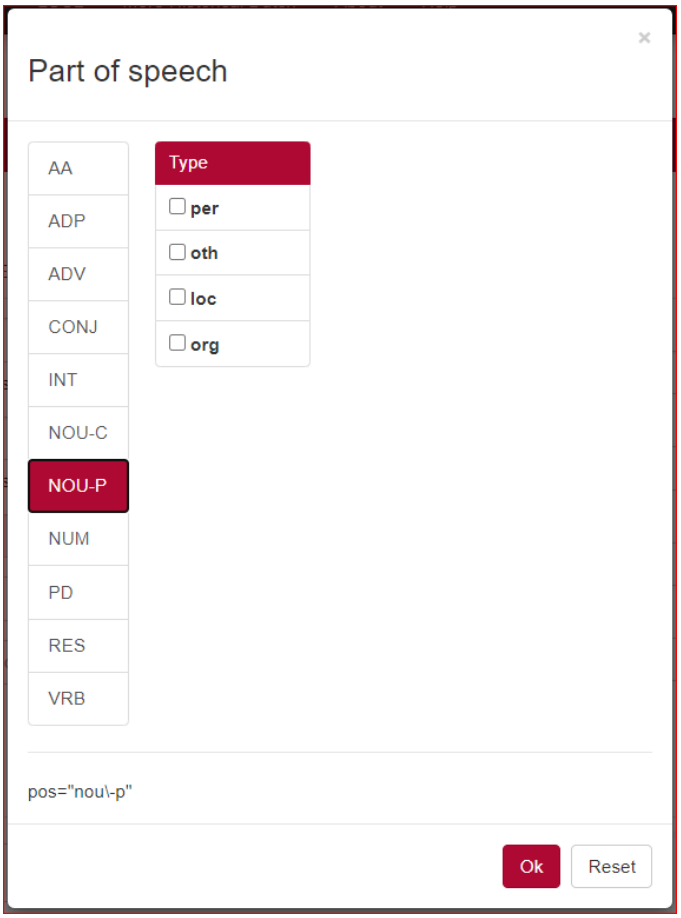
Upload a list of values

At the right sight 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 dialog box

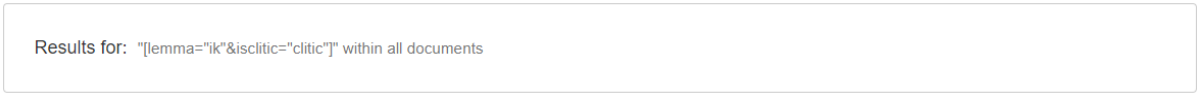
Clicking on the pencil next to the search field Part of speech provides you with the Part of speech dialog box.



For some of the categories on the left you can tick certain features to further specify your search query. By doing so you can for instance delimit your search, as shown in the above screenshot for noun proper.

Cliticity

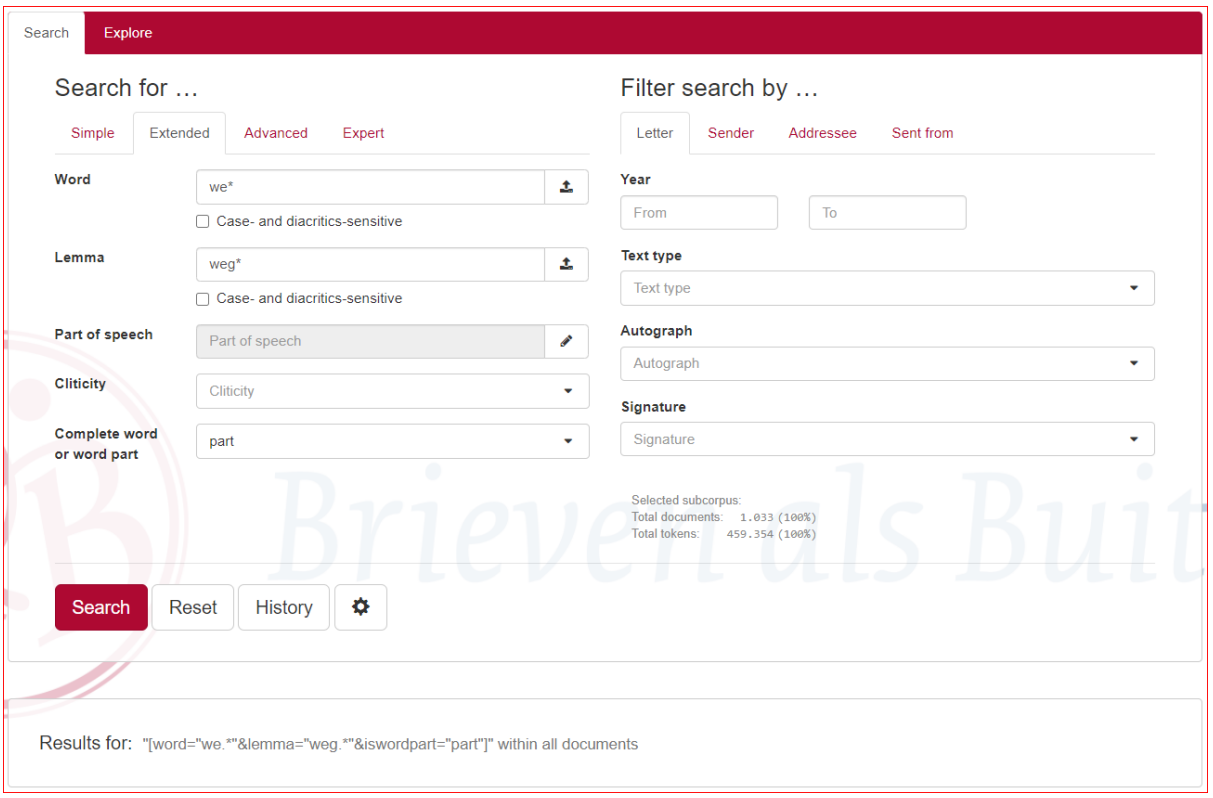
This attribute enables you to distinguish between clitical and non-clitical forms in your search. For instance, if you are interested in all clitical wordforms containing the modern lemma *ik* ('I') you should fill in *ik* at Lemma and choose clitic at Cliticity. Both search queries will be combined, as can be seen in the search query:



This search results in hits such as *ickse* (ik+ze), *soudemijn* (zullen+ik) and *ickt* (ik+het).

Complete word or word part

This option makes it possible to search for words that are split into two or more parts. Think of separable verbs as the infinitive *weglopen* ('to run away'; conjugated form *loopen ... wegh*, *liep ... wegh*) and pronominal adverbs as *daarom* (*daer + omme*, *daer + om*, *daar + om*). Keep in mind that you can only find both parts at the same time using Lemma (*weglopen*) and the option part. If you are specifically looking for just one of the composing parts (e.g. *weg*), you can enter that separate part in Word and click on the option part. In order to find all occurrences with that word part, it is necessary to take into account the different spelling variants of that word part (e.g. *weg, wegh, wech, wecht, weck, wek, weeg*). A possible search query to find all these forms is:



Starting a new search

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

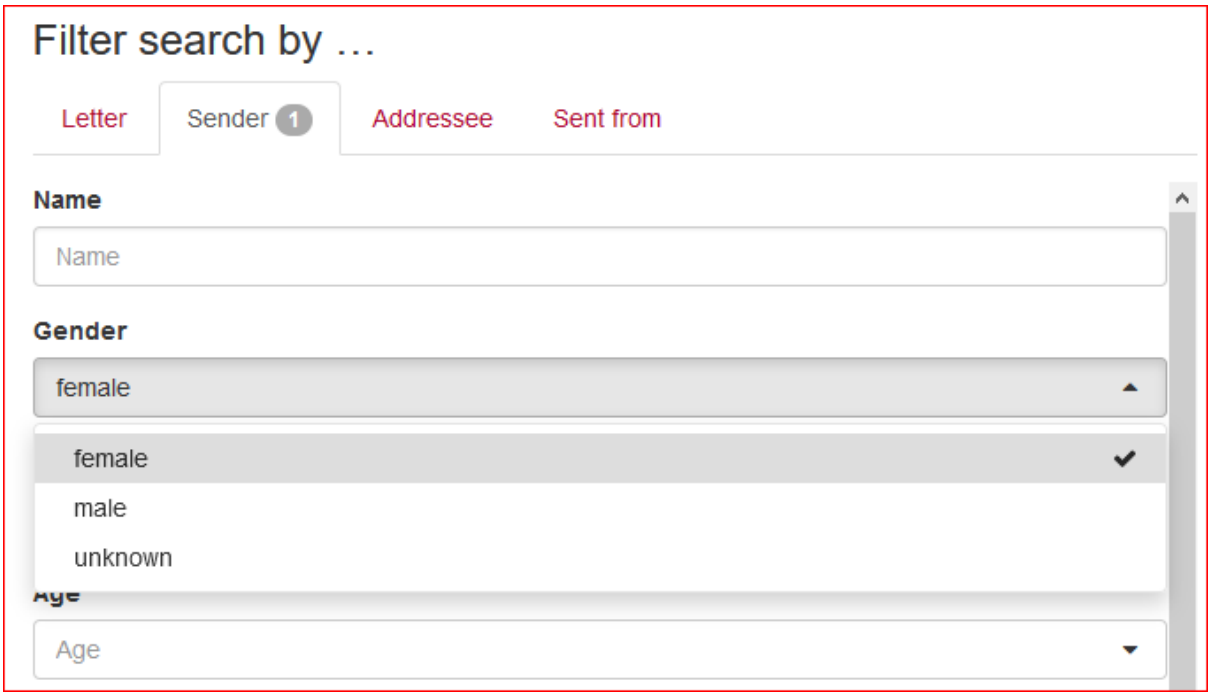
The search fields Word and Lemma are provided with a list, which contains suggestions for possible search terms in alphabetical order, based on the characters typed in.

If you only use the fields Word or Lemma, there are two possibilities to start a search: fill in the desired value and press enter, or click the Search button. The only way to start a new search after a change in Part of speech, Cliticity or Complete word or word part is to click the Search button.

Filter search by

At the right side you will find the option to limit your query to a subset of documents with specific metadata values. You can apply different filters for Letter (*Year, Text type, Autograph, Signature*), Sender (*Name, Gender, Class, Age, Region of residence, Relationship to addressee*), Addressee (*Name, Place, Country, Region, Ship*) and Sent from (*Place, Country, Region, Ship*). (To view the results for all documents simply leave the attributes in the filtering form empty.)

By means of a number at the top of Filter search by, the number of values used to filter on, is displayed:



There are two different ways to specify a filter, depending on the field type. You can either fill in a value yourself - for instance Sender Name - or choose one or more values from a drop-down list - for instance Class. The drop-down list has been applied especially when the number of values to choose from is relatively small. Sender Gender for instance has only three possibilities (female, male and unknown). 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.)

When on the other hand the set of possible values is rather large (e.g. Addressee Name), you have to type a specific value in a search field. After entering a single character, a list of possible values is suggested. Clicking on an auto-completed value will paste that value in the field. Note that this only works with a single word, like *abraham*. In order to search for an exact phrase, i.e. a multiple word value, it must be surrounded by double quotes. For instance, in the field Sender: Name *"abraham pattijn"* will result in a single letter sent by him.

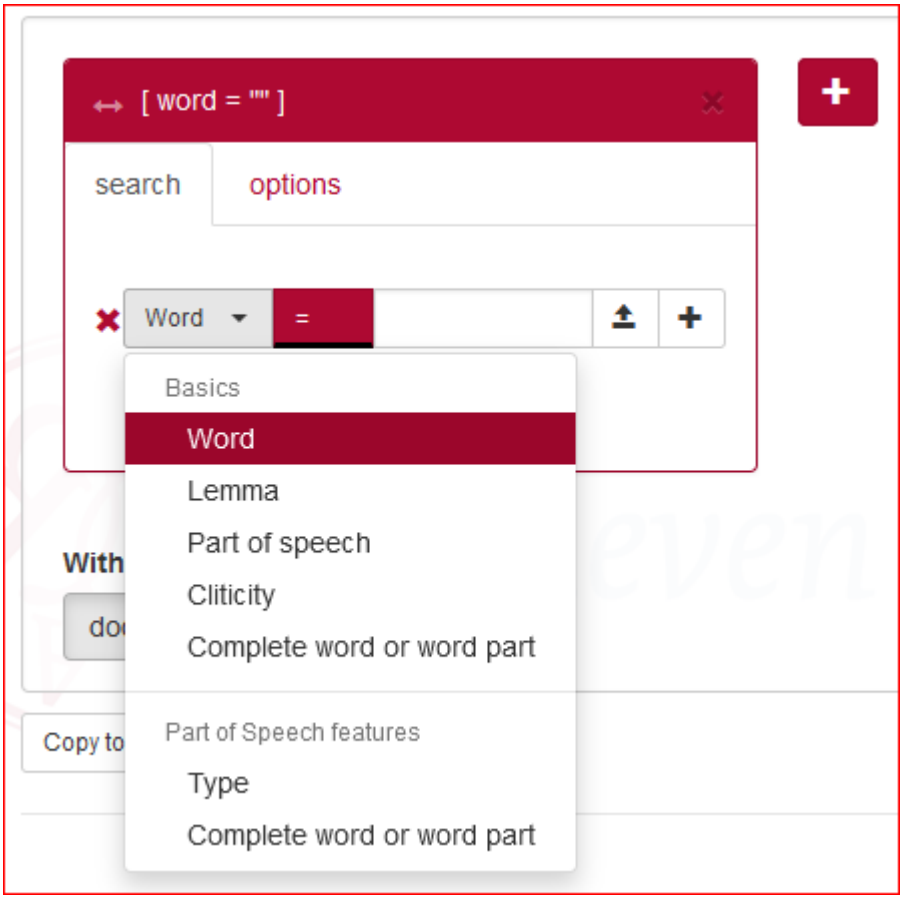
For a detailed description of the metadata, see the section Metadata categories at the beginning of this manual.

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.



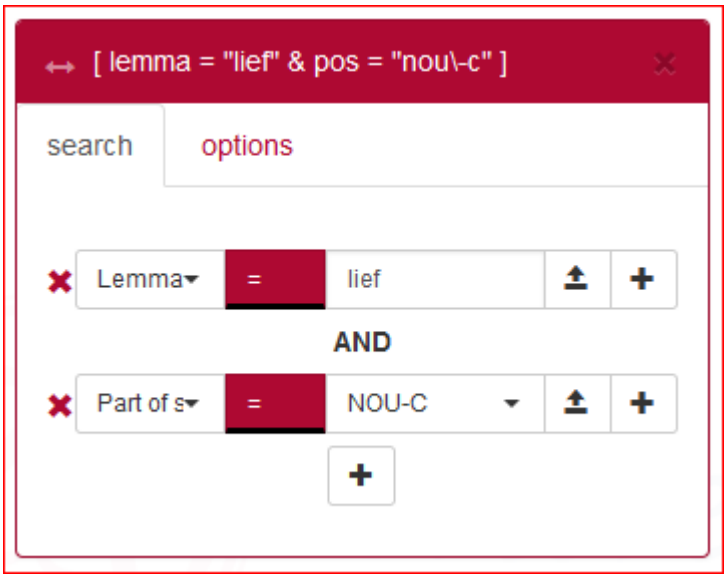
A token box in the querybuilder has two tabs: search and options.

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=NOU-C 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:



Token attributes

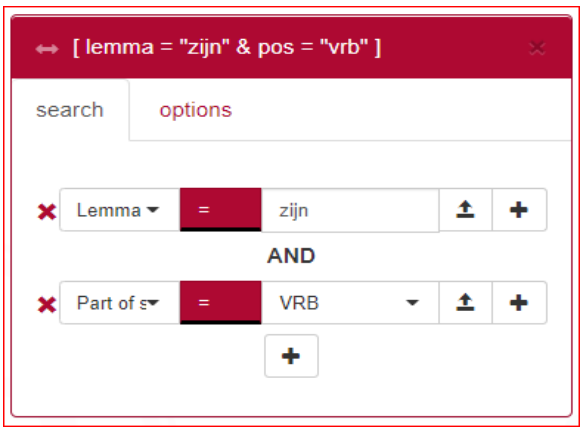
Specifying token attributes is similar to the Extended Search form. Select which attribute a token should have, and enter the value that the attribute must have for the token to be matched. Attributes in the query builder are interpreted as *regular expressions*. Note that this is different from the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



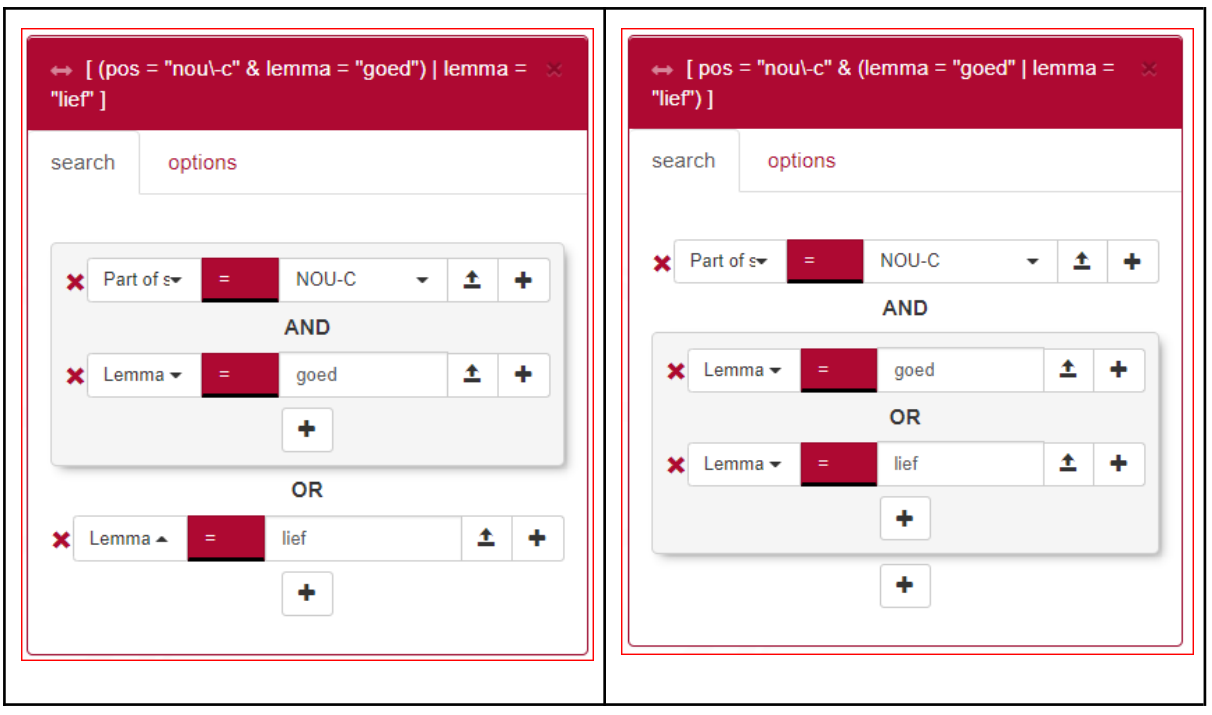
Similarly, creating a new attribute using *OR* will create a token query matching tokens that have the original attribute *or* the new attribute. For instance, enter *Word=er* first, add a new attribute with the *OR* option and enter ADV as Part of speech to match tokens with part of speech tag adverb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

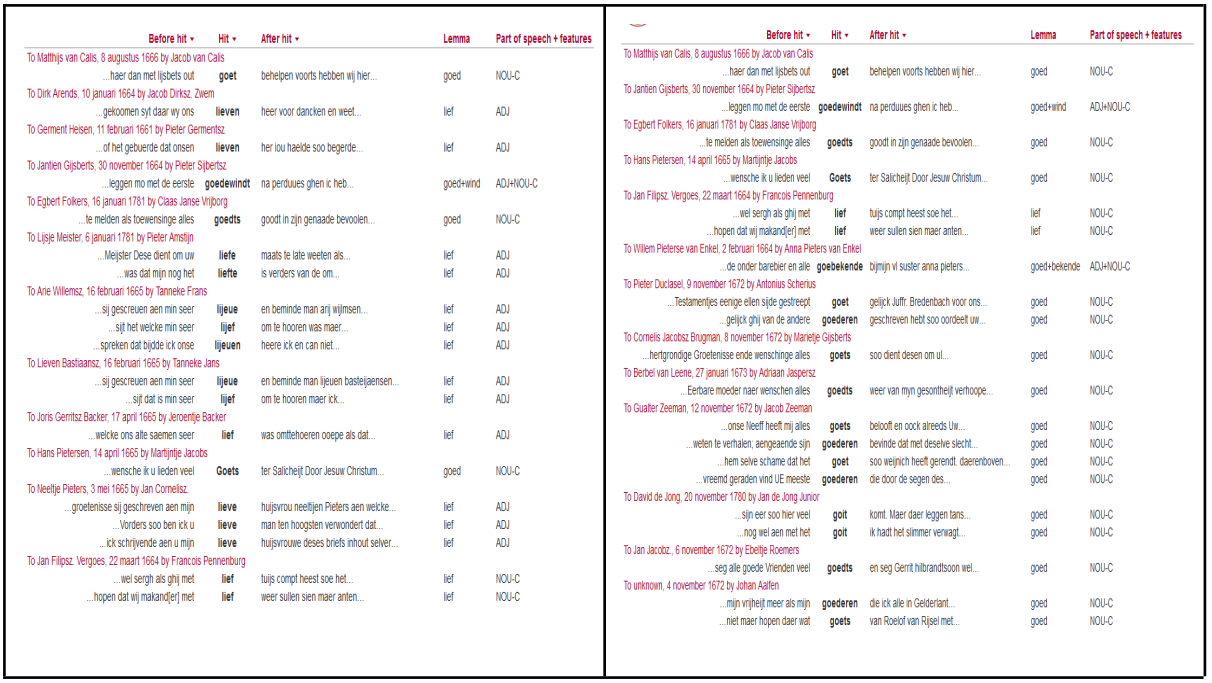
Function of the two +-buttons in a token box

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 using the +-signs **below** the attributes in the order Part of speech = NOU-C AND Lemma = *goed*, OR Lemma = *lief*, as in the left screenshot below, we get the token query: [(pos = "nou\-c" & lemma = "goed") | lemma = "lief"]. This will also match adjective forms of *lief*, as in "Seer warde en welgeagte lieffe man", where *lieffe* is an adjective, so this is not what we were after.

If, on the other hand, we add the attributes using the +-signs **right** of the attributes in the order Part of speech = NOU-C AND Lemma = *goed*, OR Lemma = *lief*, as is shown in the right screenshot below, we get the token query: [pos = "nou\-c" & (lemma = "goed" | lemma = "lief")].



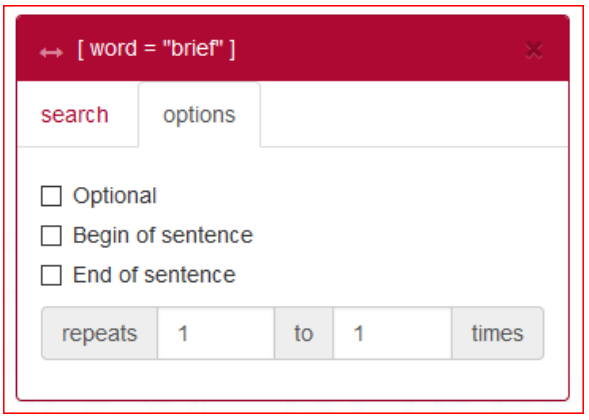


Before hit	Hit	After hit	Lemma	Part of speech + features
To Matthijs van Calis, 8 augustus 1666 by Jacob van Calis				
...haer dan met lijsbets out	**goet**	behelpen voorts hebben wij hier...	goed	NOU-C
To Dirk Arends, 10 januari 1664 by Jacob Dirksz. Zwem				
...gekoomen syt daar wy ons	**lieven**	heer voor dancken en weet...	lief	ADJ
To Germent Heisen, 11 februari 1661 by Pieter Germentsz.				
...of het gebuerde dat onsen	**lieven**	her iou haelde soo begerde...	lief	ADJ
To Jantien Gijsberts, 30 november 1664 by Pieter Sijbertsz				
...leggen mo met de eerste	**goedewindt**	na perduues ghen ic heb...	goed+wind	ADJ+NOU-C
To Egbert Folkers, 16 januari 1781 by Claas Janse Vrijborg				
...te melden als toewensinge alles	**goedts**	goodt in zijn genaade bevoolen...	goed	NOU-C
To Lijsje Meister, 6 januari 1781 by Pieter Amstijn				
...Meijster Dese dient om uw	**liefe**	maats te late weeten als...	lief	ADJ
...was dat mijn nog het	**liefte**	is verders van de om...	lief	ADJ
To Arie Willemsz, 16 februari 1665 by Tanneke Frans				
...sij gescreuen aen min seer	**lijeue**	en beminde man arij wijmsen...	lief	ADJ
...sijt het welcke min seer	**lijef**	om te hooren was maer...	lief	ADJ
...spreken dat bijdde ick onse	**lijeuen**	heere ick en can niet...	lief	ADJ
To Lieven Bastiaansz, 16 februari 1665 by Tanneke Jans				
...sij gescreuen aen min seer	**lijeue**	en beminde man lijeuen basteijaensen...	lief	ADJ
...sijt dat is min seer	**lijef**	om te hooren maer ick...	lief	ADJ
To Joris Gerritsz Backer, 17 april 1665 by Jeroentje Backer				
...welcke ons alte saemen seer	**lief**	was omttehoeren ooepe als dat...	lief	ADJ
To Hans Pietersen, 14 april 1665 by Martijntje Jacobs				
...wensche ik u lieden veel	**Goets**	ter Salicheijt Door Jesuw Christum...	goed	NOU-C
To Neeltje Pieters, 3 mei 1665 by Jan Cornelisz.				
...groetenisse sij geschreven aen mijn	**lieve**	huijsvrou neeltjen Pieters aen welcke...	lief	ADJ
...Vorders soo ben ick u	**lieve**	man ten hoogsten verwondert dat...	lief	ADJ
...ick schrijvende aen u mijn	**lieve**	huijsvrouwe deses briefs inhout selver...	lief	ADJ
To Jan Filipsz. Vergoes, 22 maart 1664 by Francois Pennenburg				
...wel sergh als ghij met	**lief**	tuijs compt heest soe het...	lief	NOU-C
...hopen dat wij makand[er] met	**lief**	weer sullen sien maer anten...	lief	NOU-C

Before hit	Hit	After hit	Lemma	Part of speech + features
To Matthijs van Calis, 8 augustus 1666 by Jacob van Calis				
...haer dan met lijsbets out	**goet**	behelpen voorts hebben wij hier...	goed	NOU-C
To Jantien Gijsberts, 30 november 1664 by Pieter Sijbertsz				
...leggen mo met de eerste	**goedewindt**	na perduues ghen ic heb...	goed+wind	ADJ+NOU-C
To Egbert Folkers, 16 januari 1781 by Claas Janse Vrijborg				
...te melden als toewensinge alles	**goedts**	goodt in zijn genaade bevoolen...	goed	NOU-C
To Hans Pietersen, 14 april 1665 by Martijntje Jacobs				
...wensche ik u lieden veel	**Goets**	ter Salicheijt Door Jesuw Christum...	goed	NOU-C
To Jan Filipsz. Vergoes, 22 maart 1664 by Francois Pennenburg				
...wel sergh als ghij met	**lief**	tuijs compt heest soe het...	lief	NOU-C
...hopen dat wij makand[er] met	**lief**	weer sullen sien maer anten...	lief	NOU-C
To Willem Pietersе van Enkel, 2 februari 1664 by Anna Pieters van Enkel				
...de onder barebier en alle	**goebekende**	bijmijn vl suster anna pieters...	goed+bekende	ADJ+NOU-C
To Pieter Duclasel, 9 november 1672 by Antonius Scherius				
...Testamentjes eenige ellen sijde gestreept	**goet**	gelijck Juffr. Bredenbach voor ons...	goed	NOU-C
...gelijck ghij van de andere	**goederen**	geschreven hebt soo oordeelt uw...	goed	NOU-C
To Cornelis Jacobsz Brugman, 8 november 1672 by Marietje Gijsberts				
...hertgrondige Groetenisse ende wenschinge alles	**goets**	soo dient desen om ul...	goed	NOU-C
To Berbel van Leene, 27 januari 1673 by Adriaan Jaspersz				
...Eerbare moeder naer wenschen alles	**goedts**	weer van myn gesontheijt verhoope...	goed	NOU-C
To Gualter Zeeman, 12 november 1672 by Jacob Zeeman				
...onse Neeff heeft mij alles	**goets**	belooft en oock alreeds Uw...	goed	NOU-C
...weten te verhalen; aengeaende sijn	**goederen**	bevinde dat met deselve slecht...	goed	NOU-C
...hem selve schame dat het	**goet**	soo weijnich heeft gerendt. daerenboven...	goed	NOU-C
...vreemd geraden vind UE meeste	**goederen**	die door de segen des...	goed	NOU-C
To David de Jong, 20 november 1780 by Jan de Jong Junior				
...sijn eer soo hier veel	**goit**	komt. Maer daer leggen tans...	goed	NOU-C
...nog wel aen met het	**goit**	ik hadt het slimmer verwagt...	goed	NOU-C
To Jan Jacobz., 6 november 1672 by Ebeltje Roemers				
...seg alle goede Vrienden veel	**goedts**	en seg Gerrit hilbrandtsoon wel...	goed	NOU-C
To unknown, 4 november 1672 by Johan Aalten				
...mijn vrijheijt meer als mijn	**goederen**	die ick alle in Gelderlant...	goed	NOU-C
...niet maer hopen daer wat	**goets**	van Roelof van Rijsel met...	goed	NOU-C

The tab options

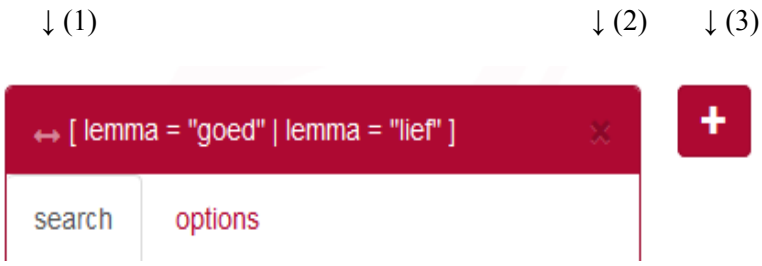
The tap options specifies the contextual properties, such as whether the token occurs at the end of a sentence, and the repetition pattern:



Managing sequences of token boxes

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

- *Rearrange* a token by clicking and dragging the little arrow handle in the top-left corner simultaneously (1).
- *Delete* a token by clicking the x in the top-right corner (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 to 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 press the Clear button on the bottom left corner of the Edit box. Using the Reset button will start a complete new search.

Copy to CQL editor

It is possible to copy a query - like [lemma="ik" & isclitic="clitic"] - to the CQL editor using the *Copy to CQL editor* button. 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 (these correspond to the token boxes in the query builder).

In CQL, spaces only affect a search if they are included in quotes. Whether the search command is [word="schip"] or [word = "schip"] does not make any difference to the result. However, there is a difference between the queries [word="schip"] and [word=" schip"]. The first search results in exactly 700 hits, but the second one in zero!

Some examples:

- Simple: `[word="schip"]`, e.g. the attribute word matches the regular expression *schip*; `[word!="schip"]`, e.g. the attribute word does **not** match the regular expression *schip*; `[lemma=".*schip"]` matches all lemmata ending with *schip*, including *schip* itself. (Note that `[lemma="*schip"]` will not give any results, because in Expert Search an asterisk is not a wildcard but a repetition operator.)
- Simple sequence: `[pos="pd"][lemma="hopen"]` matches all occurrences of the lemma *hopen* preceded by a pronoun.
- Combination of attributes (combining operators are &, |, !), e.g. `[word="hoop"&pos!="nou-c"]` (or equivalently `[word="hoop"&!pos="nou-c"]`) matches all occurrences of *hoop*, not being a noun.
- Repetition operators: `[pos="aa"]{3}` matches a sequence of 3 adjectives, `[pos="aa"]{2,4}` matches a sequence of 2 to 4 adjectives, `[pos="aa"]{3,}` matches a sequence of 3 or more adjectives.
- The empty `[]` matches any token, e.g. `[pos="aa"][]{4}[pos="aa"]` matches two adjectives with 4 arbitrary tokens in between.
- The operators |, & and parentheses () and the repetition operators (+, *, ? and {}) can be used to build complex sequence queries. Example: `"liefhebbende""man"|"almagtige""god"`, or even `("liefhebbende""man"|"almagtige""god")+`, matching any sequence of *liefhebbende man* or *almagtige god*. 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 CQL manual in the appendix, which contains further pointers.

Copy to query builder

When the query is relatively simple - like [pos="aa"][lemma="god"] - 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.

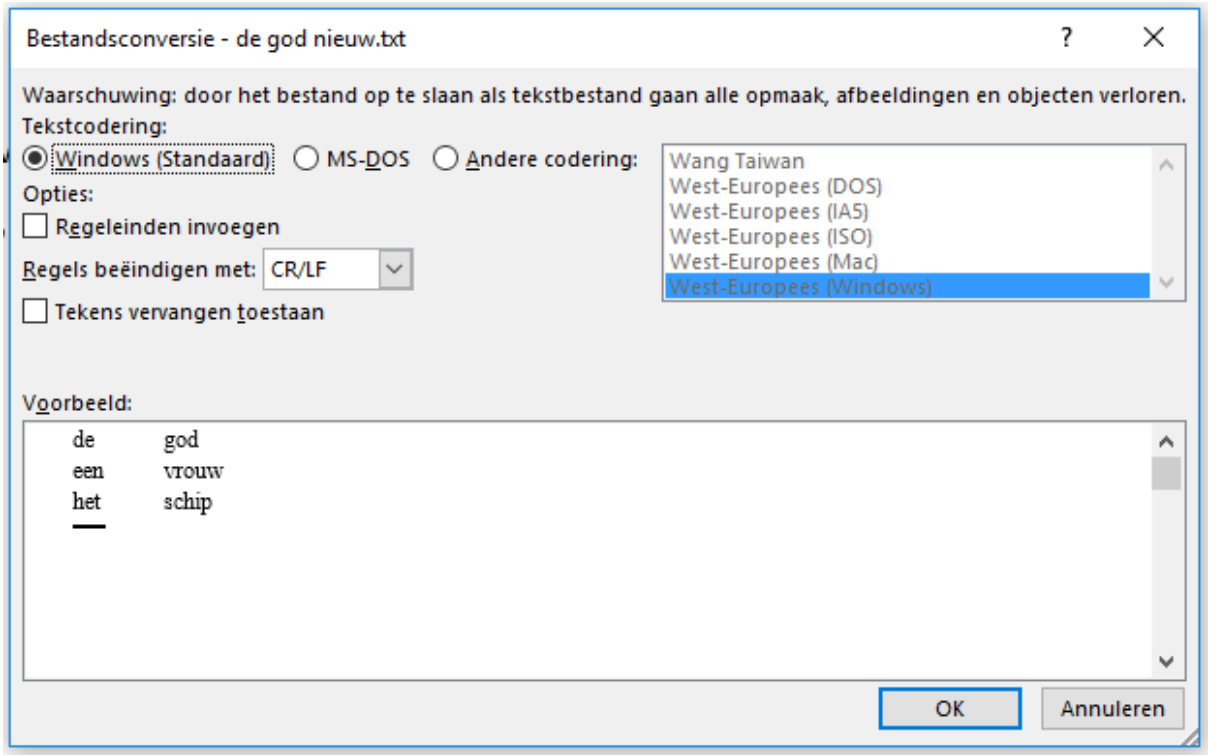
Import query

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 afterwards 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, as is shown in the following screenshot:

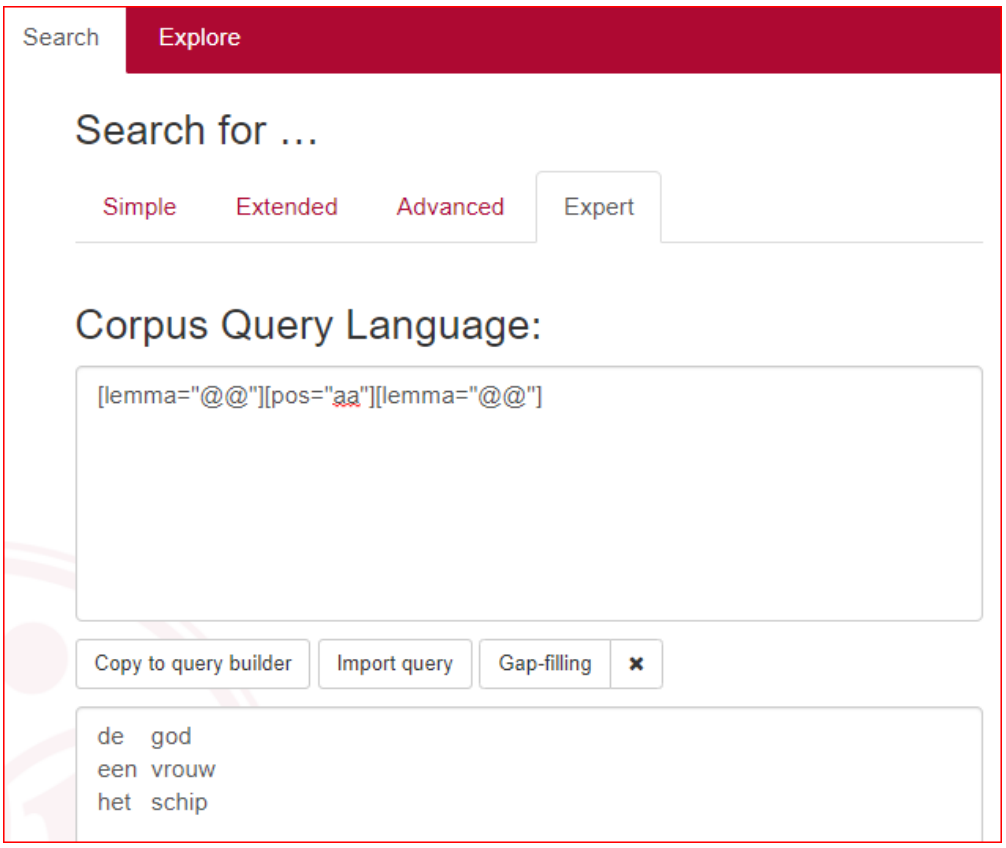


A .tsv file or a comparable .txt file enables you to complete a query with marked gaps.

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

```
[lemma="@@"][pos="aa"][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:



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 - *god, vrouw, schip* - 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 + features
…tijdt mar vor hoep dat	**de goed godt**	ons var dan wilde bijstant…	de goed God	PD(subtype=art) AA NOU-P(type=per)
…iss mij ledt mar hebbe	**den goeden godt**	te dancken wen ijck up…	de goed God	PD(subtype=art) AA NOU-P(type=per)
…mit har ambacht ijck dancke	**den goeden godt**	dat ijck noch so ene…	de goed God	PD(subtype=art) AA NOU-P(type=per)
…kompt vor hoepe nu dat	**de goede godt**	unss wilde vrede wer geuen…	de goed God	PD(subtype=art) AA NOU-P(type=per)
…al nou wij betrouw op	**de groote godt**	die hoop ijck salt voor…	de groot God	PD(subtype=art) AA NOU-P(type=per)
…te Aen hooren Dat weet	**De goeden godt**	Die en kender van Alle…	de goed God	PD(subtype=art) AA NOU-P(type=per)
…goede ghesontheijt zijn wij dancken	**de goede godt**	hier toe voor sijn ghenade…	de|De Petersen goed God	PD(subtype=art)|NOU-P(type=per) AA NOU-P(type=per)
…om daer door tekomen als	**den grooten godt**	nu onse hulper niet en…	de groot God	PD(subtype=art) AA NOU-P(type=per)
…vorder sal wesen is voor	**den grooten godt**	bekent als die met ons…	de groot God	PD(subtype=art) AA NOU-P(type=per)
…tijt foorhanden ijs soo het	**de almogende godt**	nijet en fersijet want wij…	de almogend God	PD(subtype=art) AA NOU-P(type=per)

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

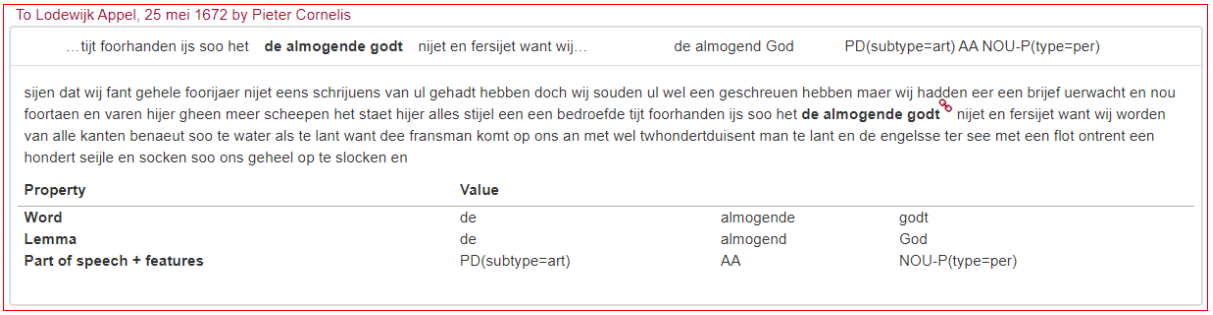
Please note that for this to work, you do need to enter @@ in the field where you want the substitution to take place. An empty field ([]) will match any term.

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 words in the column Hit - to display the properties and values of the hit (in the following example **de almogende godt)**. Click the hit again to close.

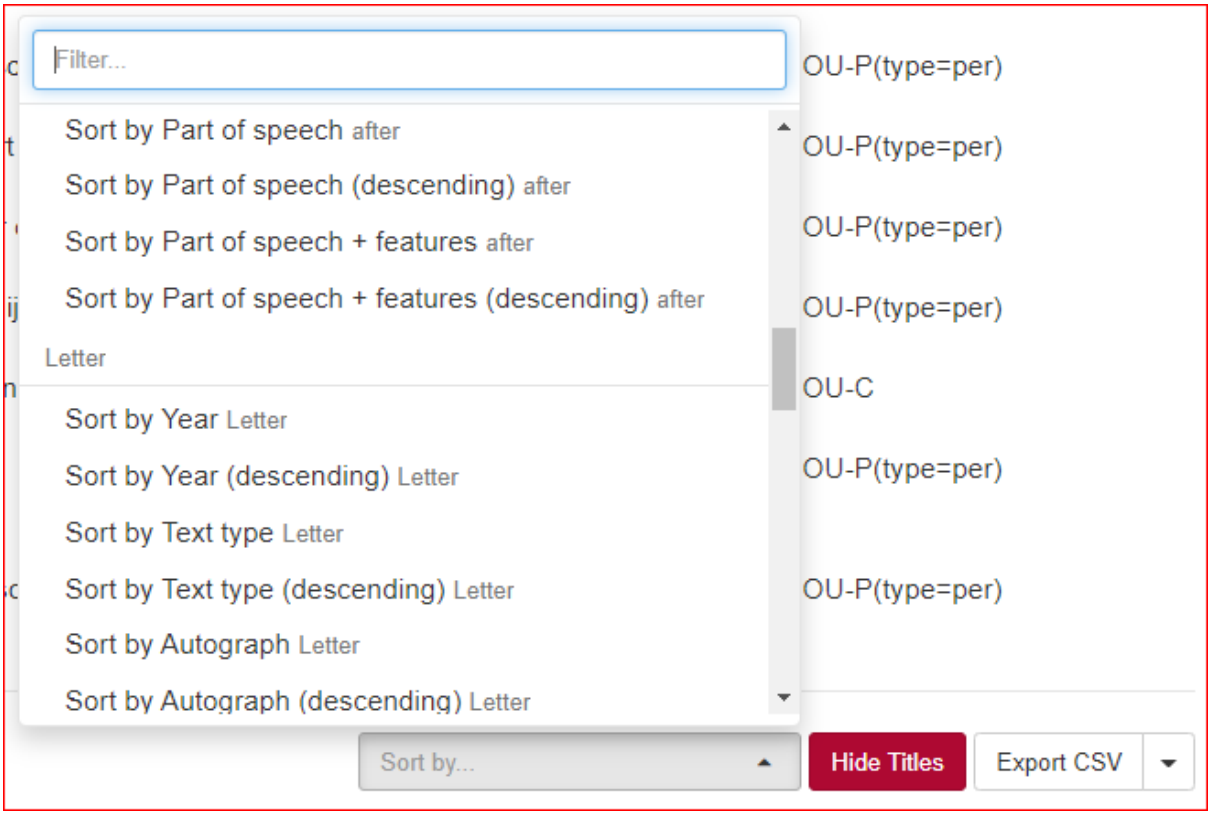


Hit rows are always preceded by a row containing the document title in which those hits occurred, in this case To Lodewijk Appel, 25 mei 1672 by Pieter Cornelis. The document titles can be toggled on or off by using the Hide Titles (or Show Titles when titles are hidden) button at the bottom of the page.

Sorting results

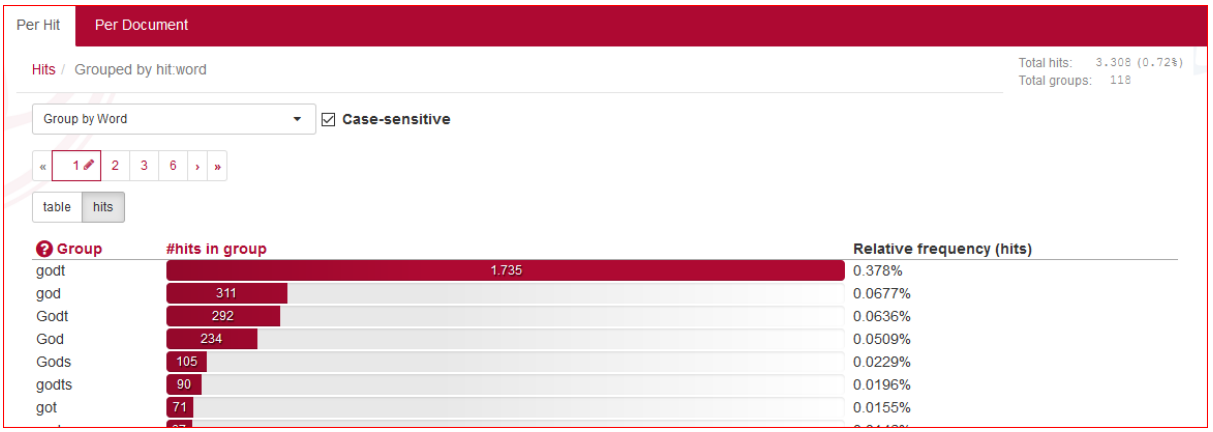
Click on any of the column headings to sort the hits on values within that column, clicking again inverts the sorting. Extra sorting options are given when clicking on Before hit, Hit and After hit: you can sort by the attributes Word, Lemma, Part of speech and Part of speech + features, as shown below.

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, by Hit, as well as by Before hit, as by After hit. Furthermore, it is possible to sort by Letter, Sender, Addressee and Sent from:



Grouping results

Results Per Hit can be grouped by properties of Hit, Before hit, After hit and by the metadata of the documents in which those hits occur (Letter, Sender, Addressee and Sent from). Grouping is facilitated by the drop-down menu Group hits by.... By selecting one of the properties a tick box appears that makes it possible to distinguish between case sensitive and case insensitive.

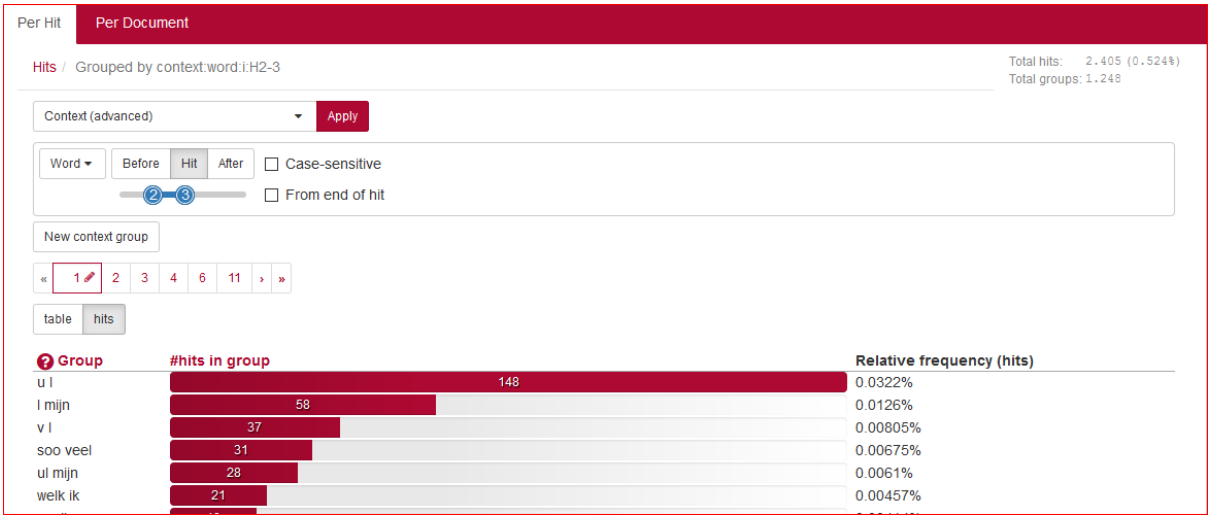


In the Per hit view, advanced grouping options are available by selecting the option Context (advanced). This option allows you to group the results by up to 5 tokens before or after the hit. It also allows you to group the results based on (parts of) the hits. By pressing the New context group you can group the results by another property or another range.

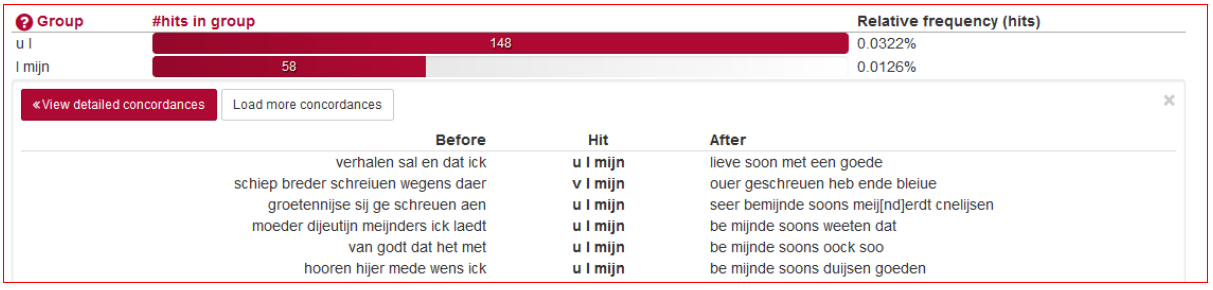
We will work that out using an example. A search for groups of three pronouns - in Expert Search: [pos="PD"]{3} - produces hits like the following (Titles are hidden):

Before hit	Hit	After hit	Lemma	Part of speech + features
…soo korts aff is is	**datmen d ene**	dach noch wel een weinch…	dat+men de een	CONJ+PD(subtype=oth) PD(subtype=art) PD(subtype=oth)
…om te hooren voorts laet	**ic u mijn**	alderliefste huijs vrou Janten gijberts…	ik u mijn	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…gesond ben hoopen de van	**uw het selven**	was het anders het souw…	u hetzelfde hetzelfde	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…ouwe jaar gekoome wy wensen	**jou ale veul**	seege int nuwe jaar veul…	jij al veel	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…breek ik of en wens	**uw alle des**	heeren seegen en geesond heyd…	u al de	PD(subtype=oth) PD(subtype=oth) PD(subtype=art)
…het moogelijk slim afgeloopen hebben	**het welke geen**	eer voor harmanij is tog…	hetwelk hetwelk geen	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…en gezondt ben hoope van	**uw alle het**	zelve te hoore als wij…	u al hetzelve	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…gezondt ben hoope van uw	**alle het zelve**	te hoore als wij in…	al hetzelve hetzelve	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…standig hede te schrijven hoop	**ik uw self**	haast te spreeke en ik…	ik u zelf	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…tot reeseele wel geerueert sijt	**het welcke min**	seer lijef om te hooren…	hetwelk hetwelk mijn	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…nalate eens te scrijuen dat	**ghij soo veel**	gelijeft te doen na de…	gij zoveel zoveel	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)
…order En ik hoope dat	**gij deese mij**	ne ge ringe letter ook…	gij deze mijn	PD(subtype=oth) PD(subtype=oth) PD(subtype=oth)

It is now possible to group the hits by the second and third tokens of those hits. See below.



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.

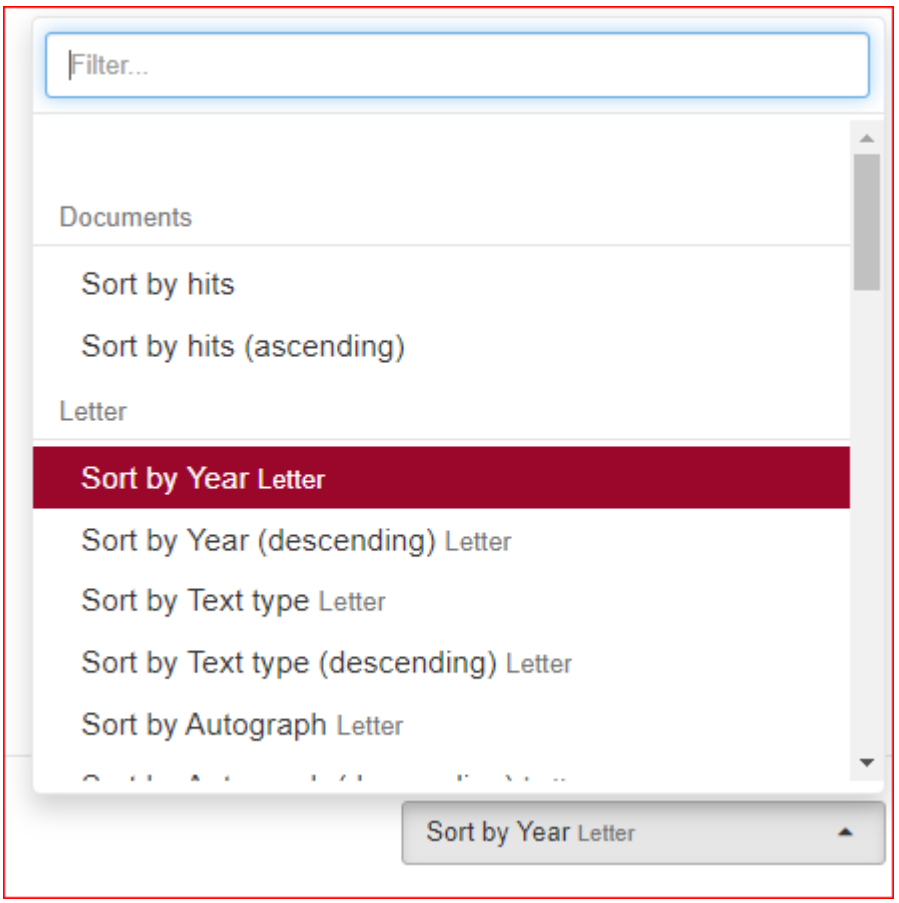


Click on View detailed concordances to go back to the normal hits view to see more detailed information for the hits in this group. The button Go back to grouped view brings you back to the list of groups.

Per Document view

Sorting results

Results can be sorted by means of the drop-down menu at the bottom of the page, which enables you to sort on Documents (e.g. the number of hits for your search query) and on the metadata of the Letter, the Sender, the Addressee, Sent from.



Grouping results

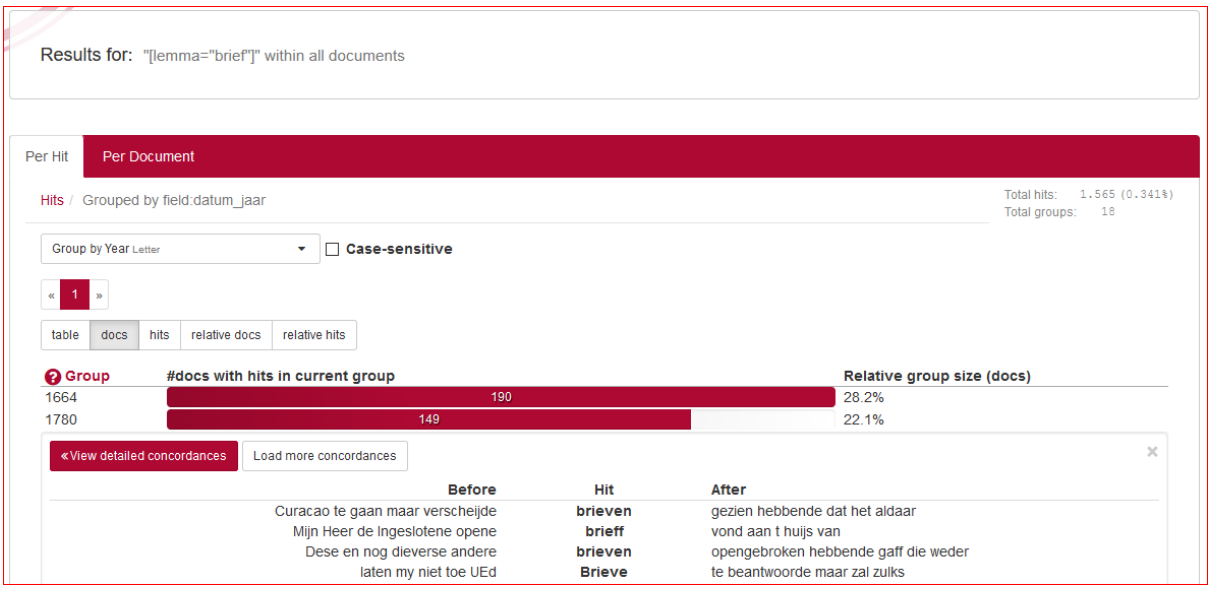
Results Per Document can be grouped by the metadata of the documents in which those hits occur (Letter, Sender, Addressee, Sent from). Here, grouping is facilitated by the drop-down menu Group docs by....

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 red bar of a specific group and then on View detailed concordances (see screenshot below). 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 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. You can navigate from one hit to another by using the arrows at the Hits button:

When you hover with your mouse over a specific word in the document - for instance *weeke* - 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:

Metadata of a document

In the Metadata tab all metadata properties of the document are displayed. They provide information about the Letter, the Sender, the Addressee and Sent from.

Statistics

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

Images

Under images you can find a photo of the original letter, kept in the National Archives (Kew, UK).

Exploring the corpus

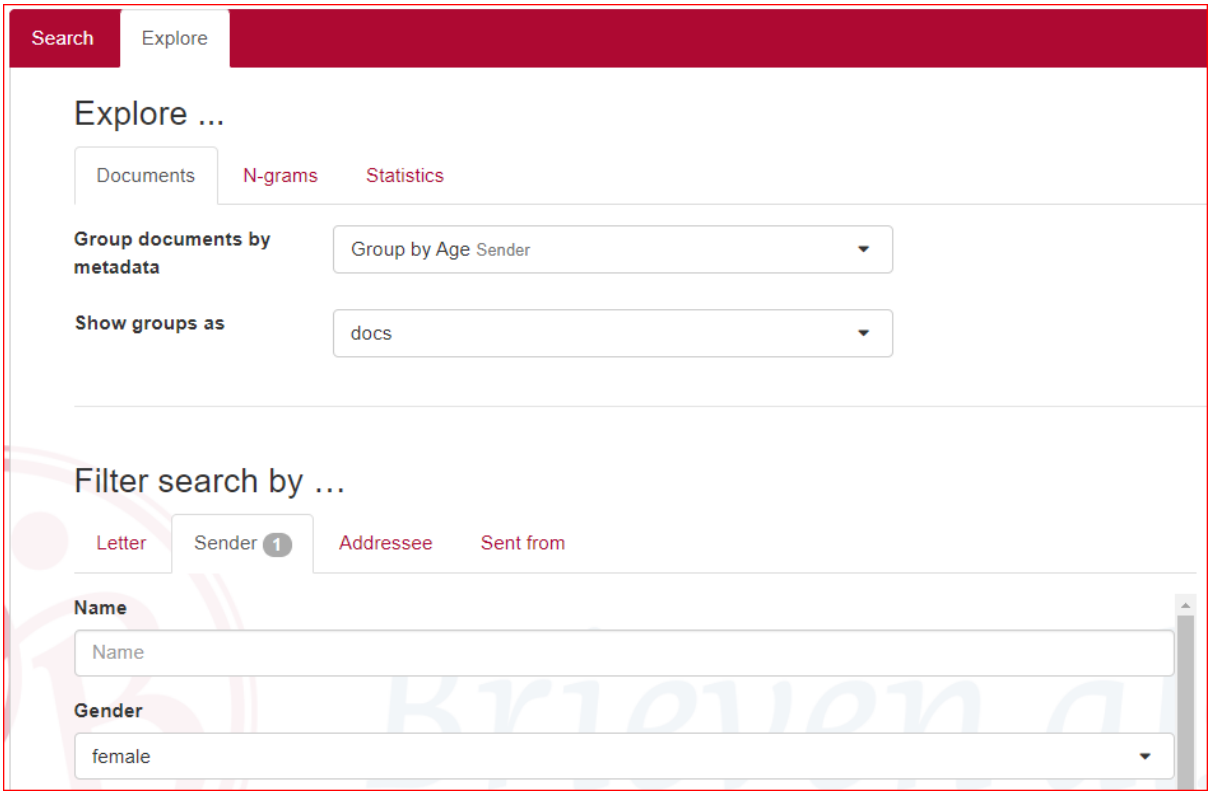
The Explore tab has three subdivisions: Documents, N-grams and Statistics.

Documents

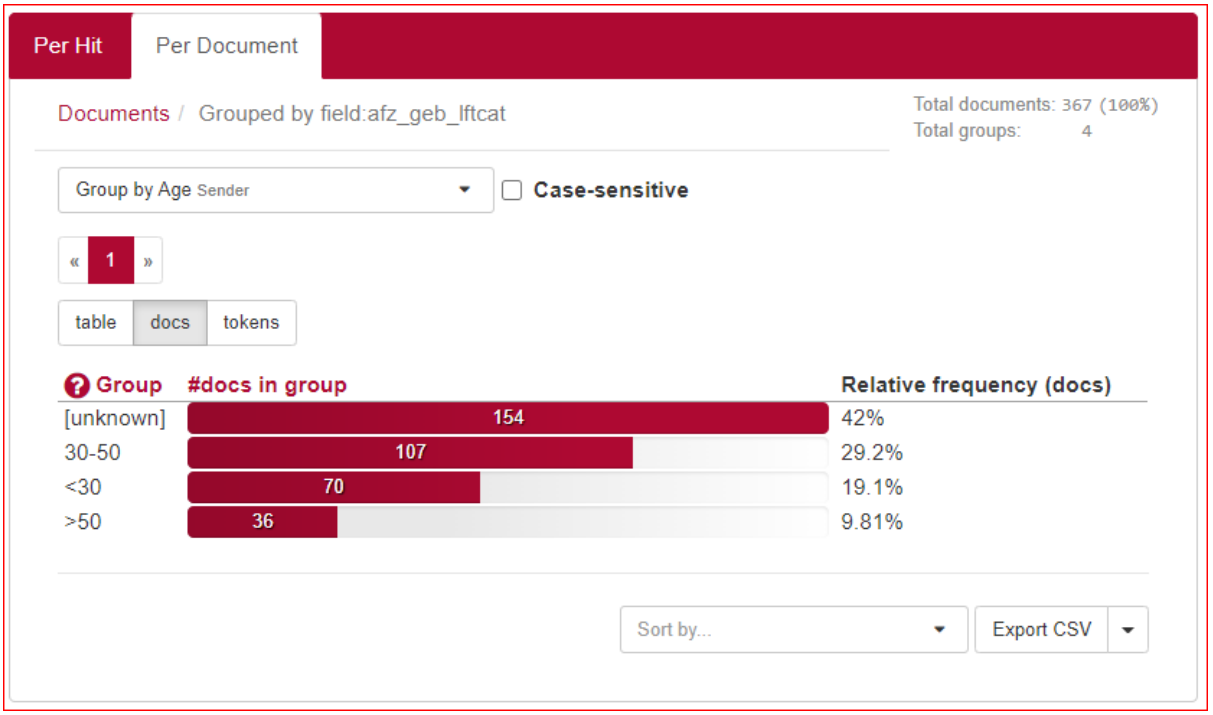
This subtab allows you to investigate the documents. It consists of two drop-down menus to specify the grouping of the metadata and to specify the way the groups are to be shown.

A simple example: suppose we want to obtain information about the age distribution of female senders within the *Brieven als Buit* corpus.

- In the Group documents by metadata drop-down menu, choose Group by Age (Sender)
- In Show groups as, select *docs*
- In the metadata search form (Filter search by), select in Sender Gender *female*
- Press Search



You will get this result:



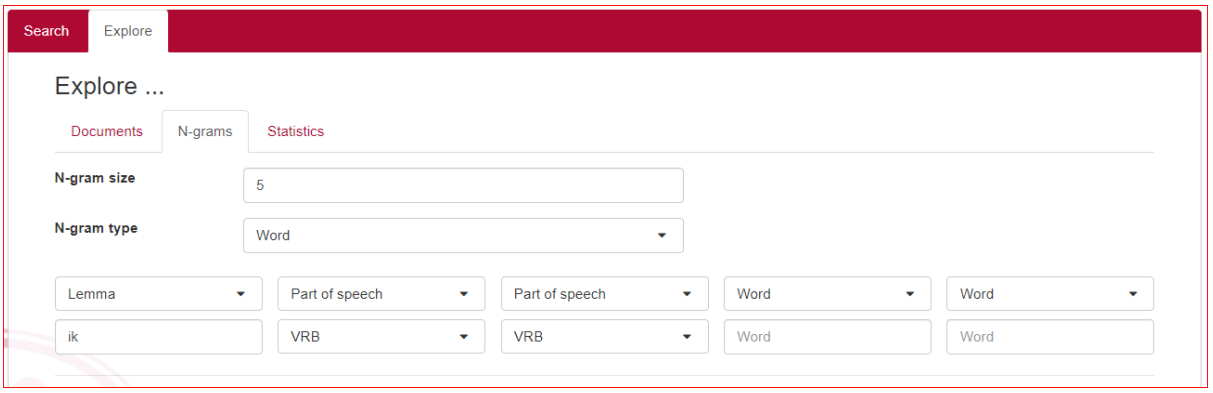
N-grams

An *N-gram* is a sequence of *N* items: Word, Lemma and Part of speech. This option will list the frequency of different N-grams in a (sub-)corpus.

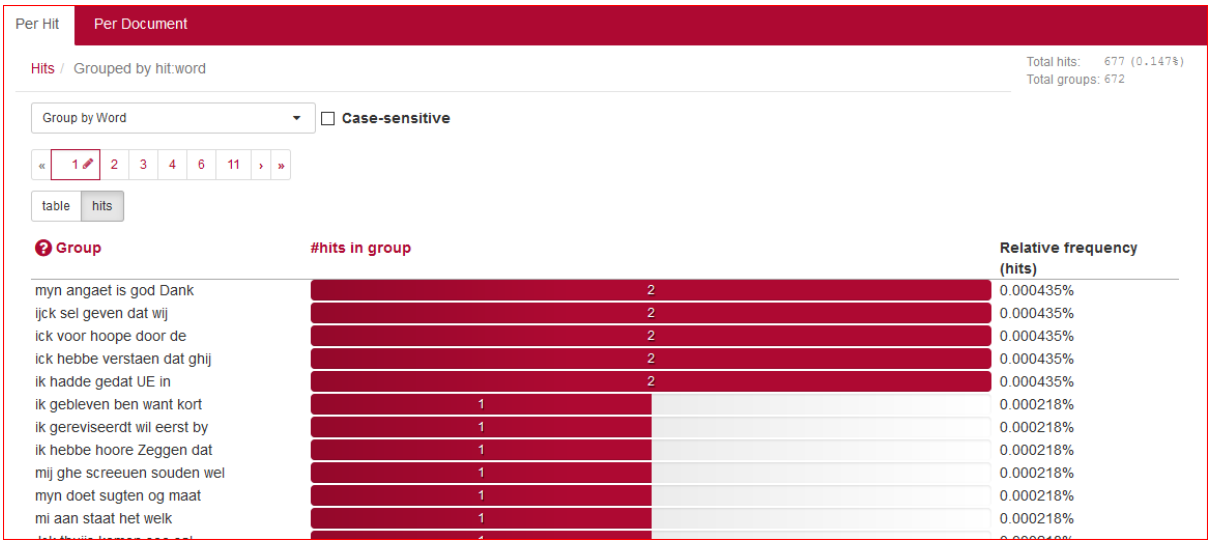
Options

- *N-gram size*: the length of the sequence (a number from 1 to 5; default setting is 5)
- *N-gram-type*: choose for sequences of Word (i.e. word form), Lemma or Part of speech. If you do not specify the search term further, a series of five consecutive Words, Lemmas or Parts of speech will be searched for.
- It is also possible to restrict to, for instance, 5-grams with some slots already specified, as is shown in the following example.
- By using the Filter search by … you can create a subcorpus within the *Brieven als Buit* corpus for specific metadata.

Example



Within all the documents of the *Brieven als Buit* corpus, you will find more than 600 occurrences of this so-called 5-gram:



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.

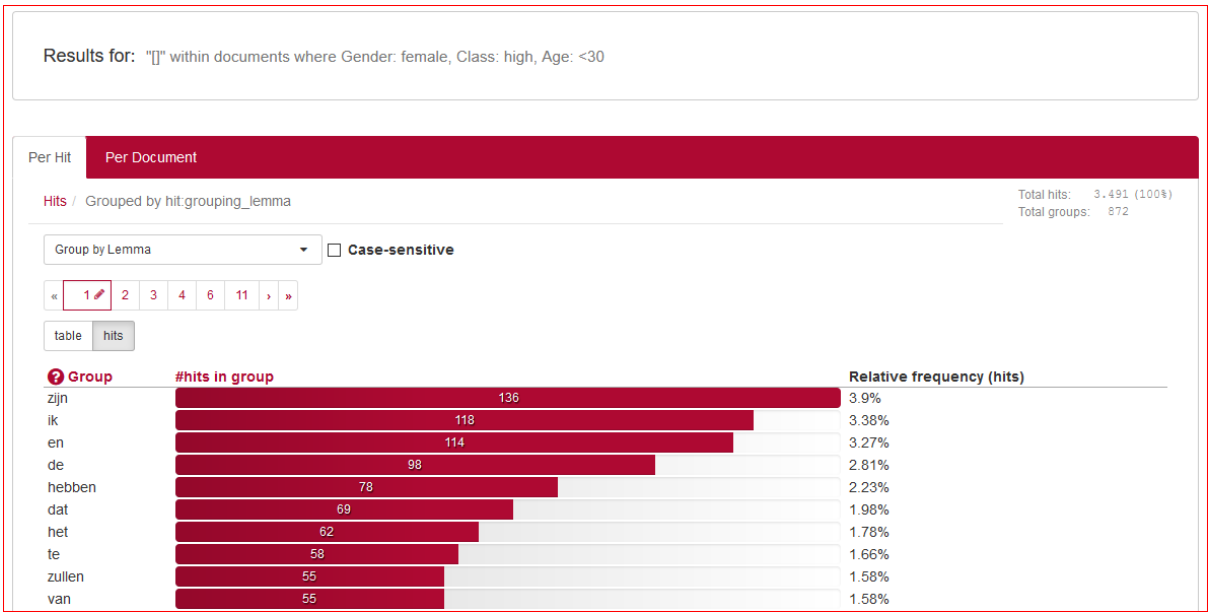
Options

- *Frequency list type:* choose for lists of Word (i.e. Word form), Lemma, Part of speech and Part of Speech + features
- By using the Filter search by... you can create a subcorpus within the *Brieven als Buit* corpus for specific metadata.

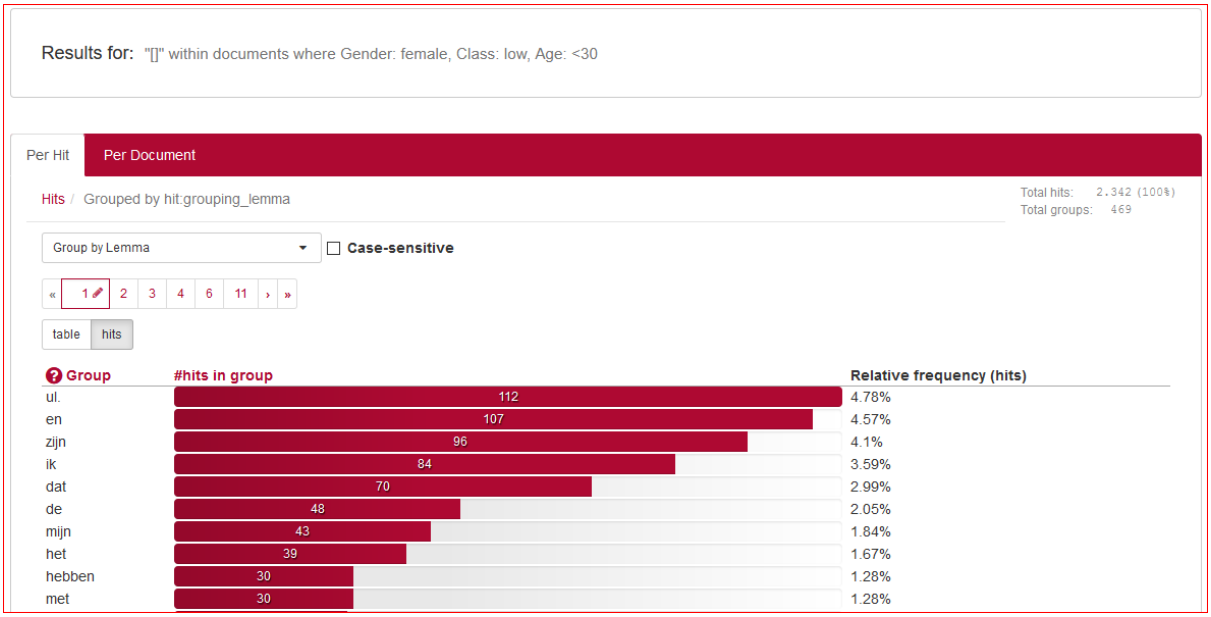
Example

It is possible to determine the use of the ten most frequent words by female writers by searching for Frequency list type Word and by filtering search by Gender, Class and Age.

For female writers who are younger than 30 years old and who belong to the high class, this results in:



For female writers who are younger than 30 years old but who belong to the low class, this results in:



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 in 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 CQP 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 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/accent-sensitive matching. Note that, unlike in CWB, case-INsensitive matching is currently the default. To explicitly match case-/accent-insensitively, 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 a 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 allow 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!" (A note about 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.)

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 that forward slash at the end of the tag. This way of referring to the element means "the whole element". Compare 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.
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.