Korpusomat EU

Release 0.1

IPI PAN

Aug 19, 2023

CONTENTS

Korpusomat is a web app for building multi-layered annotated corpora, which can then be accessed via the MTAS browser. The annotation is performed using two state-of-the-art multilingual sets of programming tools: spaCy and Stanza. The main goal of the app is to provide researchers – who do not necessarily possess any special technical skills or knowledge – with the results of operations conducted using these tools on any given (set of) text(s).

Korpusomat processes txt files as well as other major formats (e.g. epub, mobi, doc, rtf, and pdf; the complete list of supported formats is available at http://tika.apache.org/1.17/formats.html). Since the tools it employs require UTF-8 encoding, files using a different character encoding (e.g. ISO-8859-2 or CP-1250 for Polish), will be automatically converted into UTF-8.

Texts may also be added directly from the web, in which case the selected websites will be processed using the newspaper library. For details, see https://newspaper.readthedocs.io/.

CHAPTER

ONE

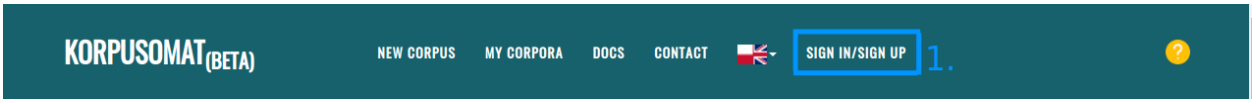
DOCUMENTATION

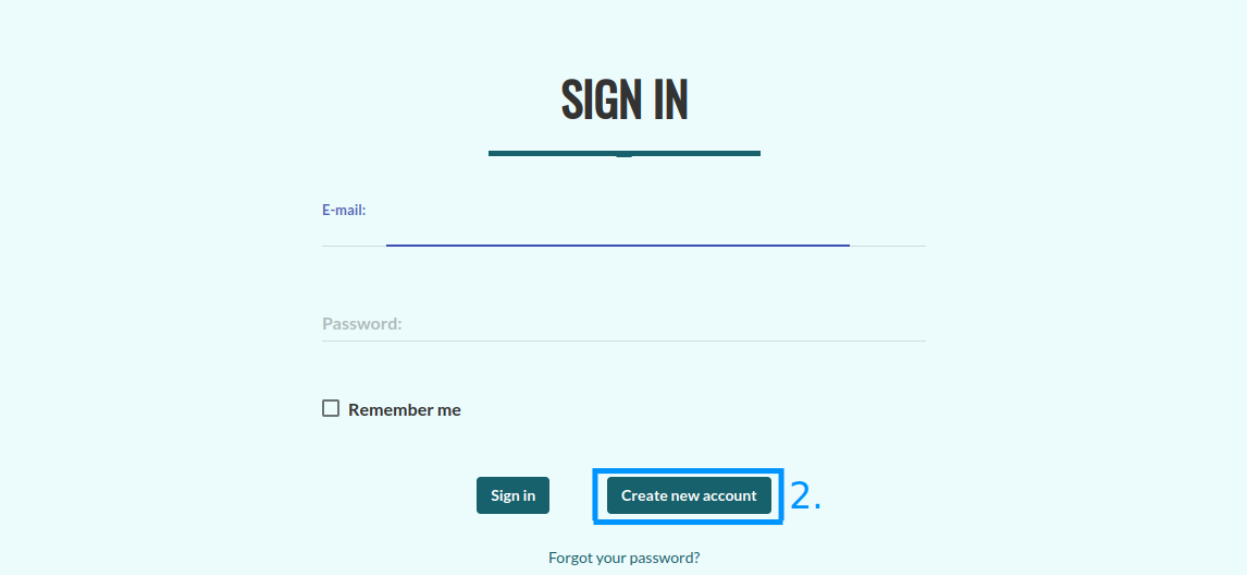
1.1 User manual

1.1.1 Creating an account

To start using Korpusomat, you need to create an account first. Creating an account is simple: just enter your email address and password.

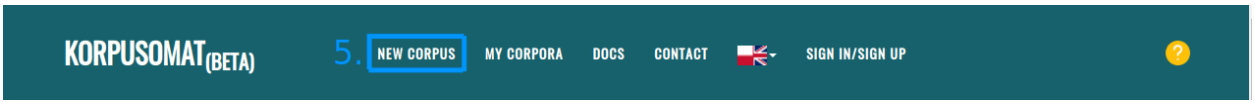
To create an account, press the Sign in/ Sign up button in the top right corner.



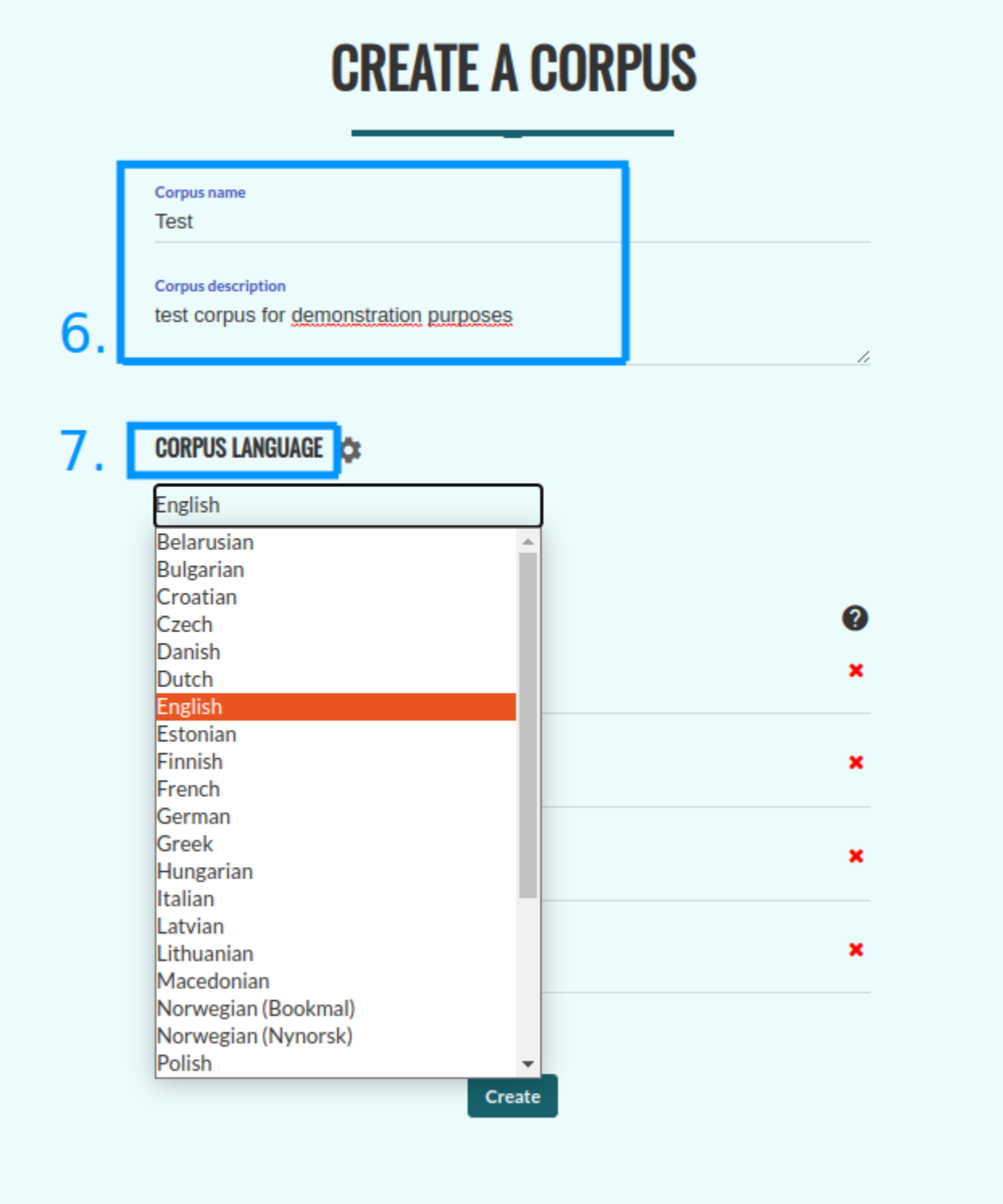


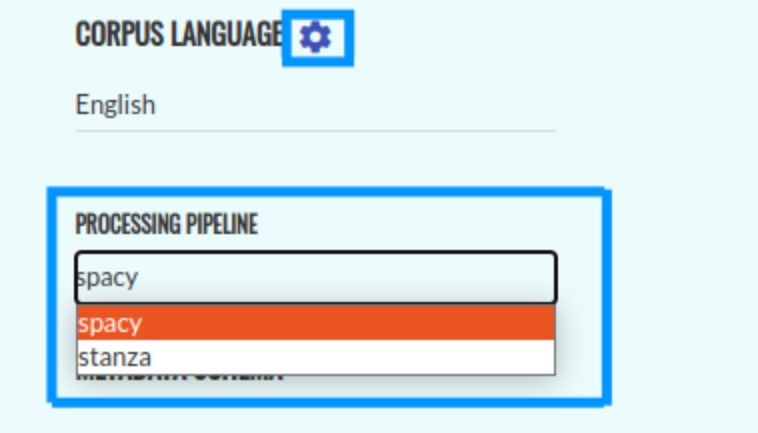
1.1.2 Creating a corpus

In order to create a new corpus, select “New corpus” from the main menu.

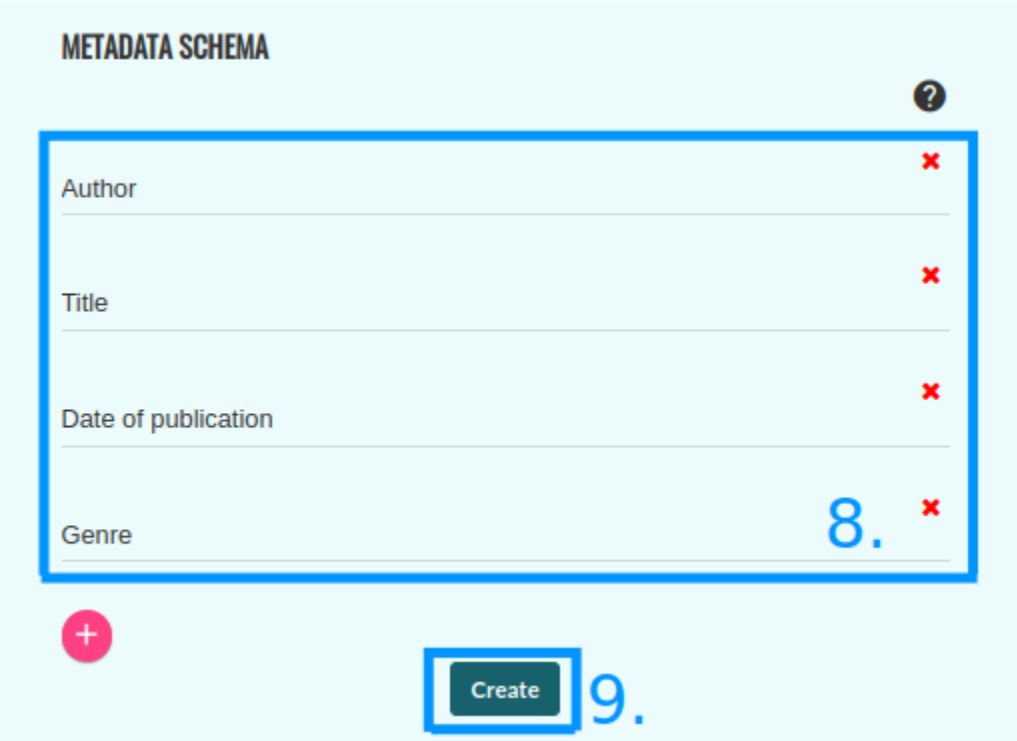


Now, enter the name and description of your corpus (6) and choose the language of the corpus (7).

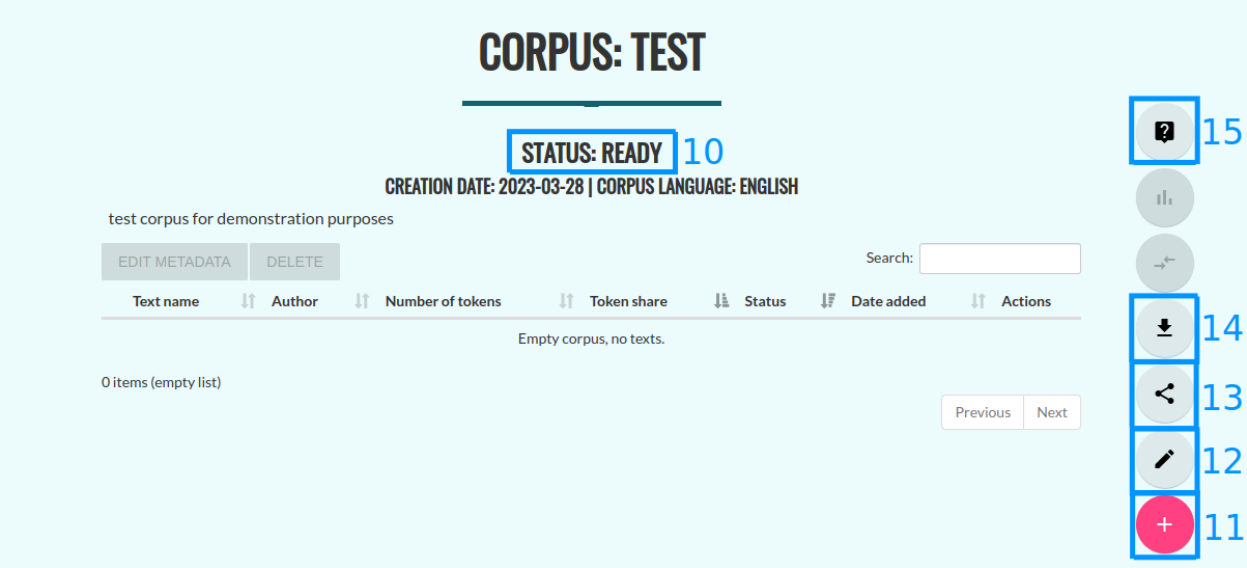

CREATE A CORPUS
Corpus name
Test
Corpus description
test corpus for demonstration purposes
6.
7.
CORPUS LANGUAGE
English
Belarusian
Bulgarian
Croatian
Czech
Danish
Dutch
English
Estonian
Finnish
French
German
Greek
Hungarian
Italian
Latvian
Lithuanian
Macedonian
Norwegian (Bookmal)
Norwegian (Nynorsk)
Polish
Create



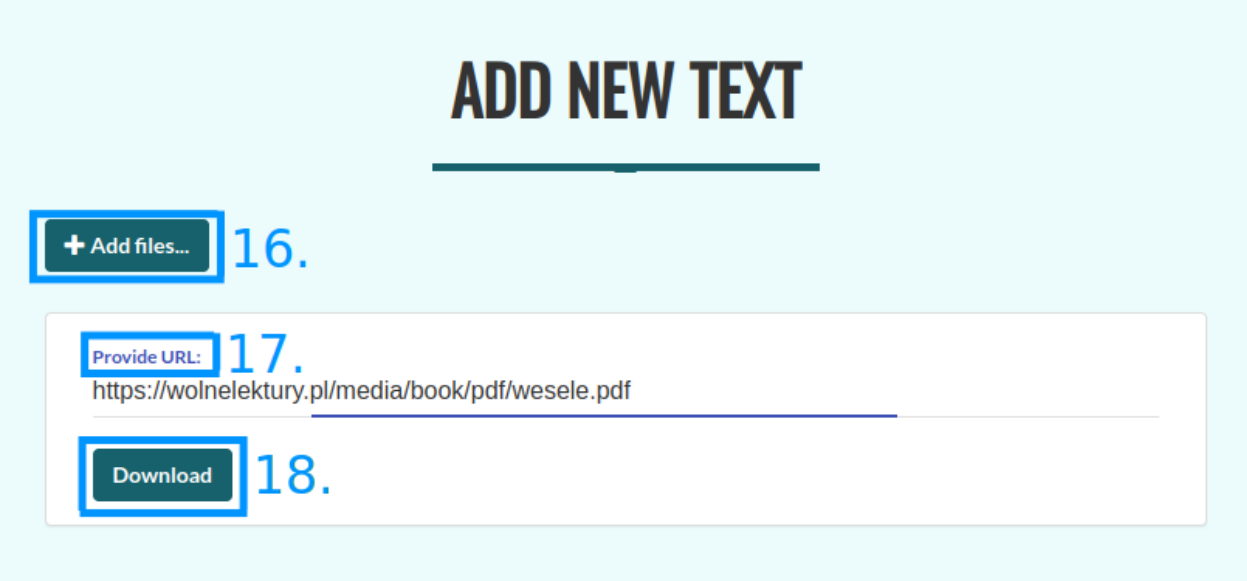
In order to select the processing pipeline, click on the settings icon next to (7). If both pipelines are available for your corpus, the default setting is Spacy. You can add metadata of every text in your corpus (8). You can also modify them once the corpus is created. To save your corpus, click on the "Create" button (9).



Next, you will be redirected to the control panel for your corpus. Once its status switches to "Ready" (10), you can start adding texts by pressing the "+" button (11).



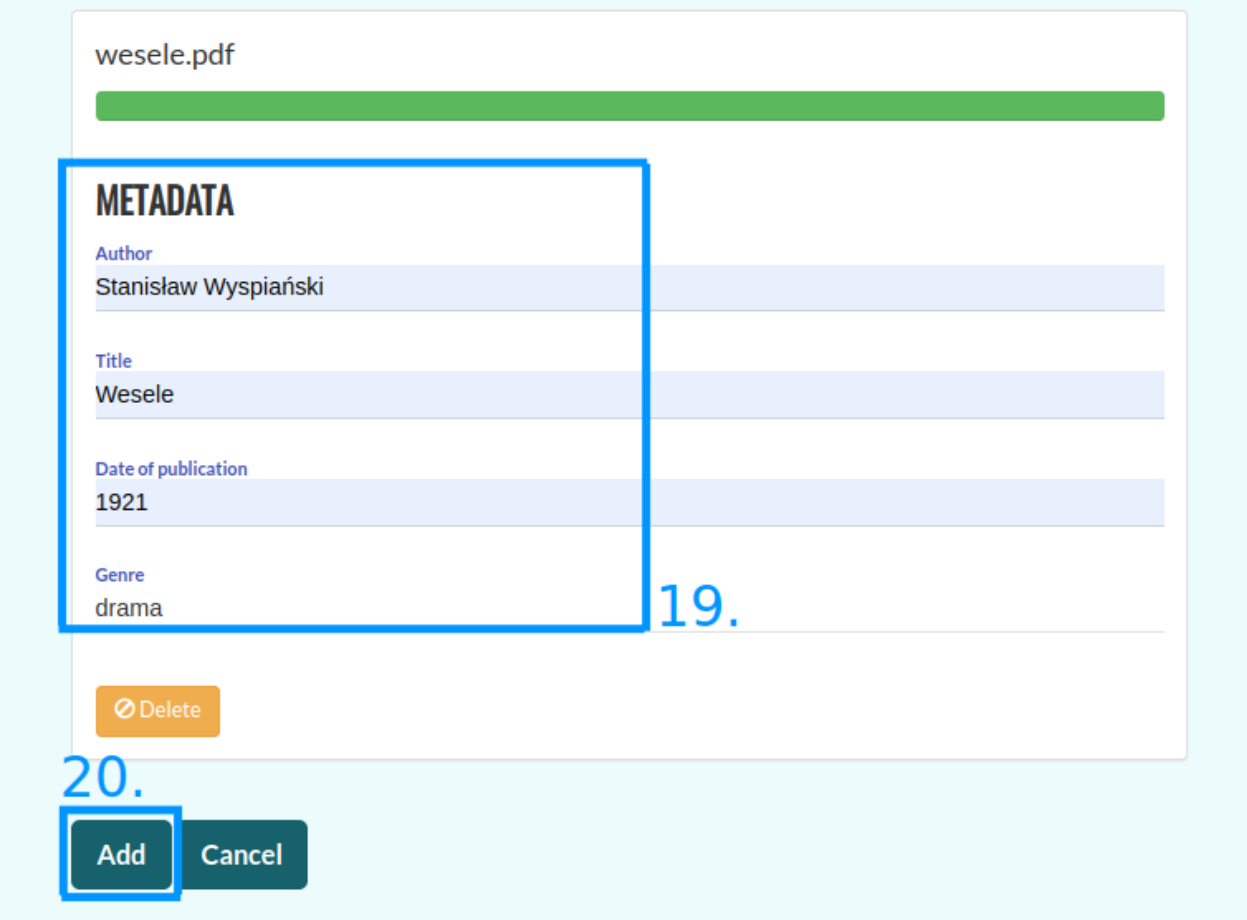
When you click on the "+" button, you will be redirected to the page which will allow you to add your texts. The list of supported formats is available here. There are two ways of adding new texts to a corpus.



One approach is to upload files from your local drive. In order to do that, you need to press the "+ Add files" button (16) and choose one or more files from the pop-up window. The other approach consists in providing links to the texts that are supposed to make up your corpus in the appropriate window (17), and clicking on the "Download" button (18). Korpusomat will download and process them automatically.

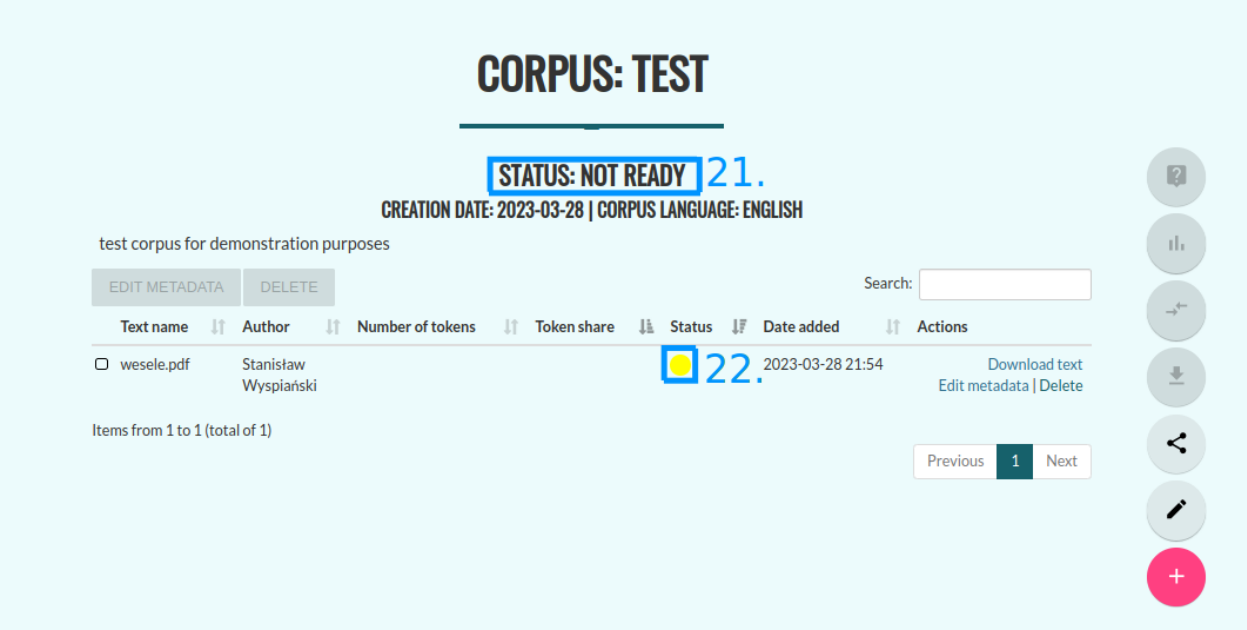
Po załadowaniu treści można wprowadzić lub zmodyfikować metadane (19). Korpusomat automatycznie spróbuje wydobyć metadane z dodanego pliku, jednak nie zawsze jest to możliwe. W wypadku tekstów w formatach EPUB i MOBI oraz stron internetowych Korpusomat spróbuje wydobyć metadane z nagłówków dokumentów. W wypadku plików tekstowych automatyczne rozpoznawanie metadanych wymaga tego, by nazwy plików zapisane były w następującym formacie: „autor - tytuł (miejsce, rok)". Przykładowo, aby Korpusomat automatycznie rozpoznał metadane „Pana Tadeusza" z nazwy pliku, dodany plik powinien nazywać się „Adam Mickiewicz - Pan Tadeusz (Paryż, 1834).txt".

To add more texts, press the "+ Add files" button again (16) or copy and paste another link in the search bar (17).

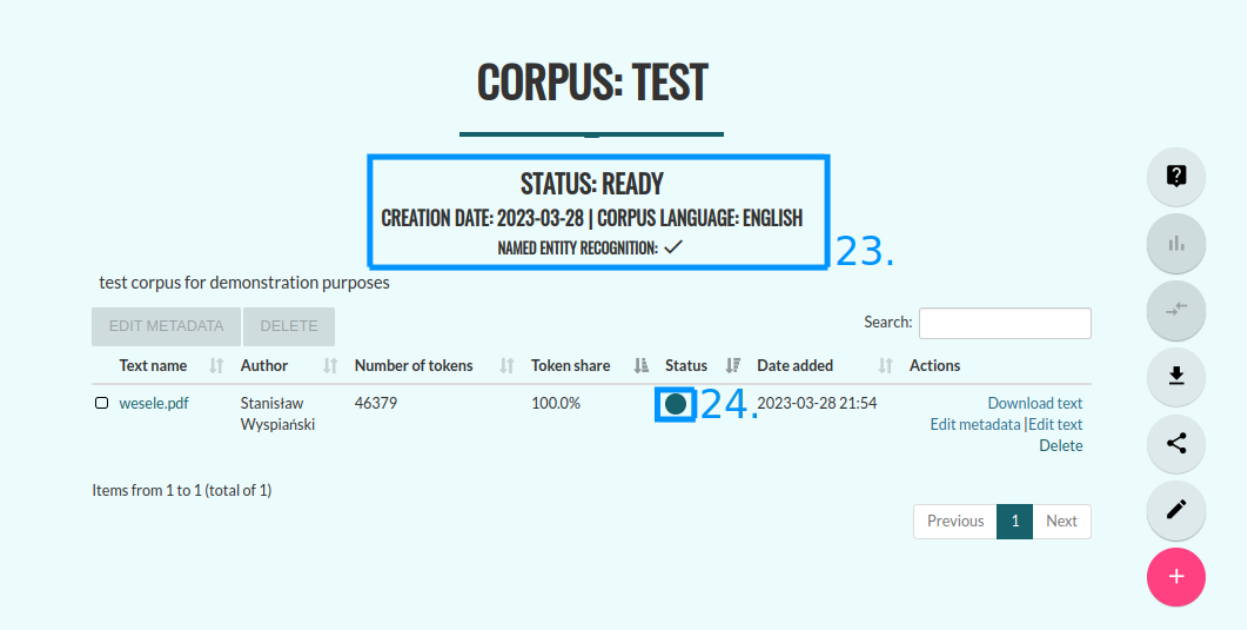


Press the "Add" button to finish adding texts to your corpus (20).

You will then be redirected back to the corpus control panel, while Korpusomat processes the added texts. You will be able to verify the status of each text, such as "In processing" (22). To process a book consisting of 80-100 thousand words, Korpusomat will need approximately 4-5 minutes, although the exact processing time depends on the server load and the selected layers of annotation. Currently, the maximum processing time of one file is 10 minutes - longer tasks will fail. While your texts are being processed, you can add new files using the "+" button (11).



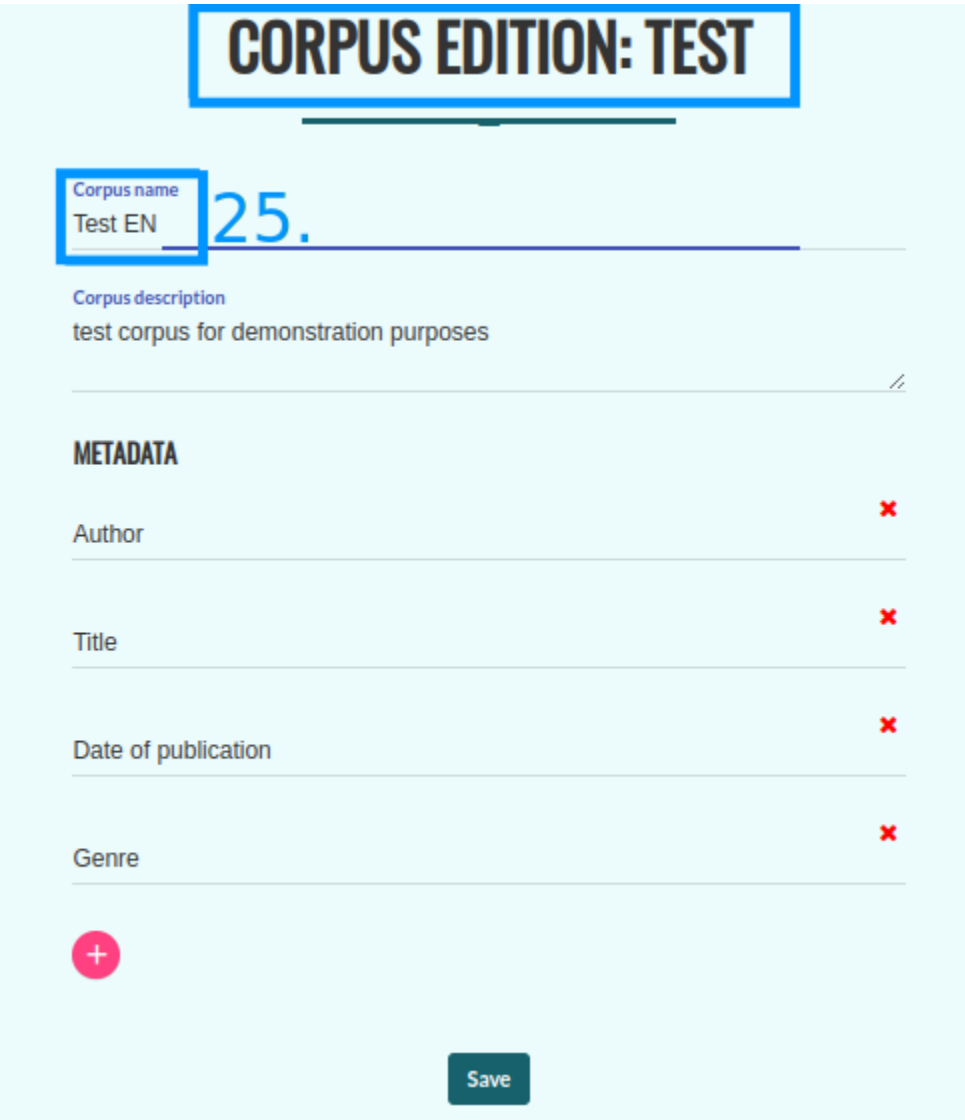
Once your texts are processed, their status will change to "Processed correctly". The status of the corpus will also then switch to "Ready" (23).



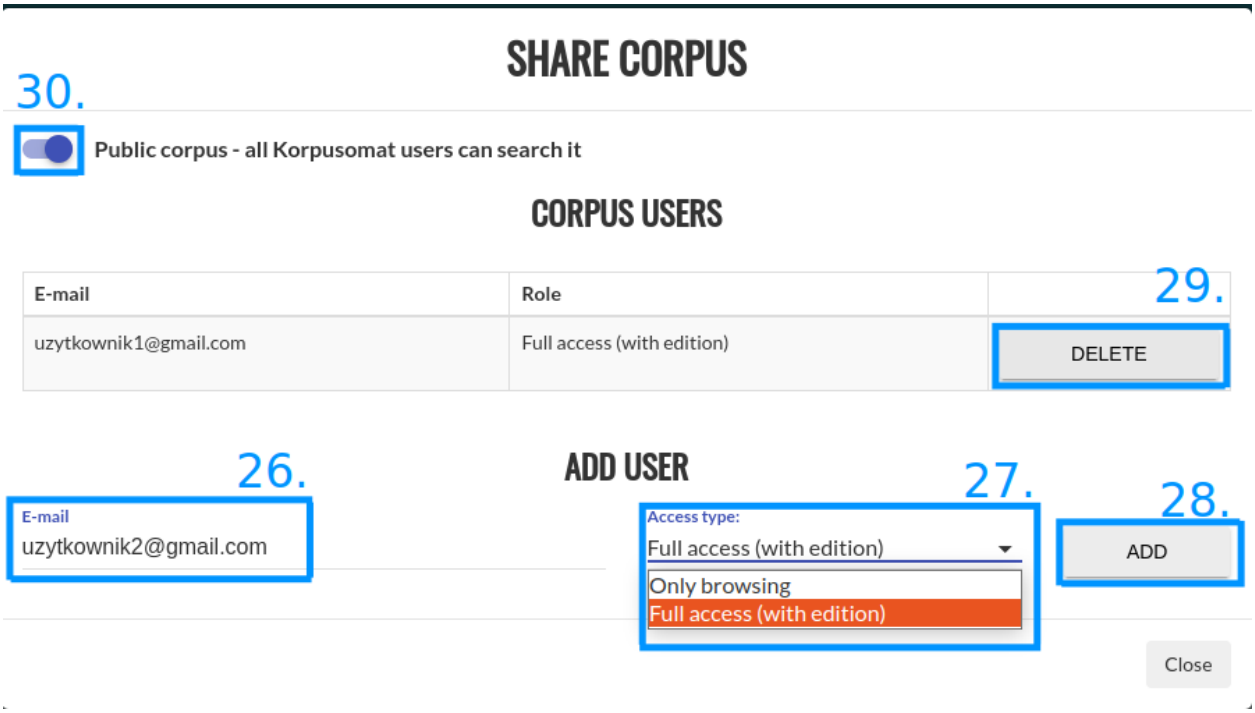
Now you can start working with your corpus. The potential next steps are:

1. Editing the corpus (12)
2. Sharing the corpus with other users (13)
3. Downloading the processed XML files (14)
4. Querying the corpus (15)

You can edit the name and description of your corpus, add and edit metadata (25) by pressing the pencil icon (12).



You can share your corpus with other users by pressing the "Share corpus" button (13). You just need to enter their email address(es) (26), choose the access type (27), and press "Add" (28). You can also remove their access to your corpus using the "Delete" button (29). In order to share the corpus with all Korpusomat users, swipe the "Public corpus" button to the right (30).



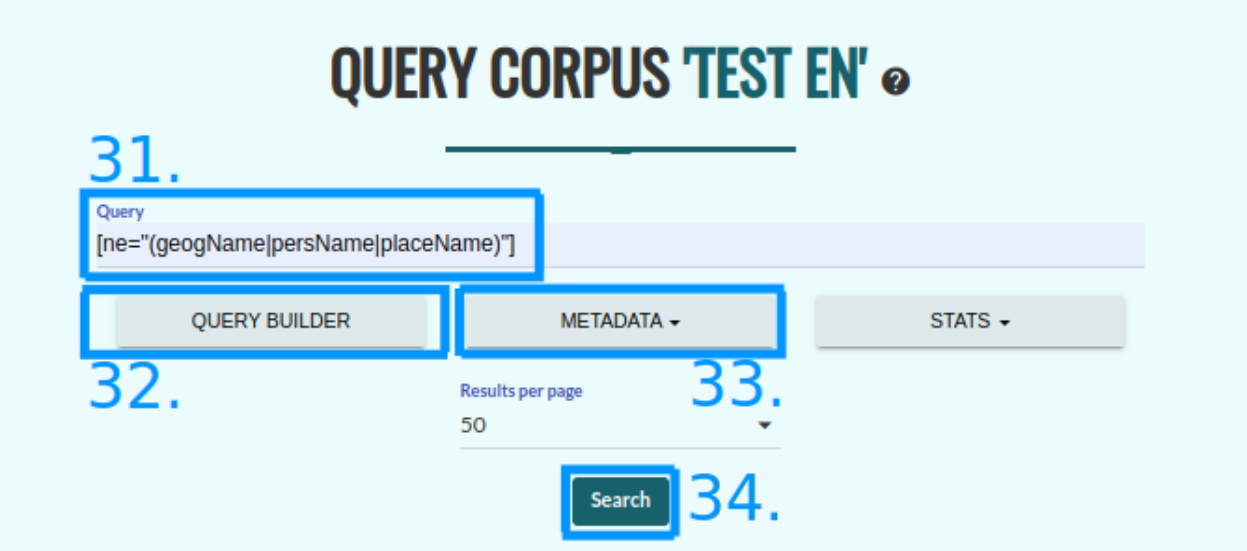
You can download the processed XML files of all the texts in the corpus by clicking on the "Download corpus files" button (14). The format of the files will be in line with the following specification CCL.

You can continue to edit your corpus. Adding or removing texts will require the corpus to be processed again.

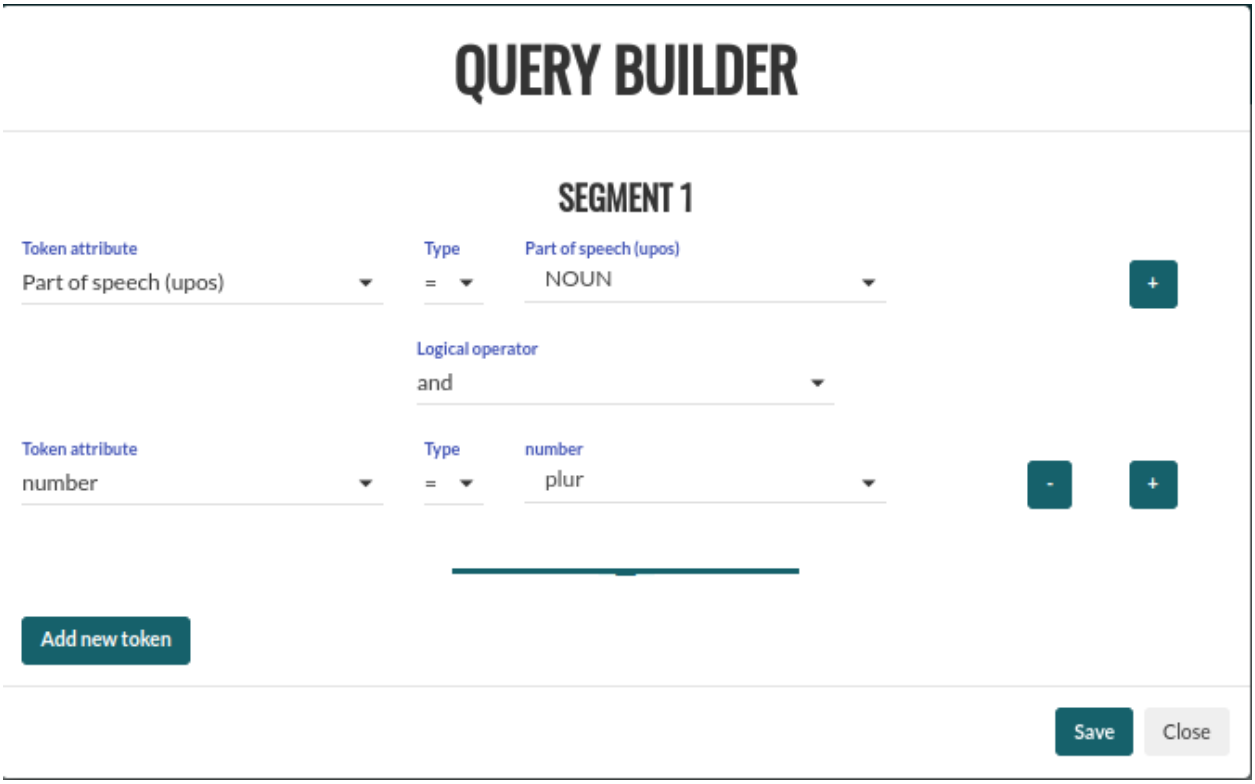
Clicking on the question mark icon (15) will redirect you to the search screen. The following sections of this guide explain how to create corpus queries.

1.1.3 Corpus querying

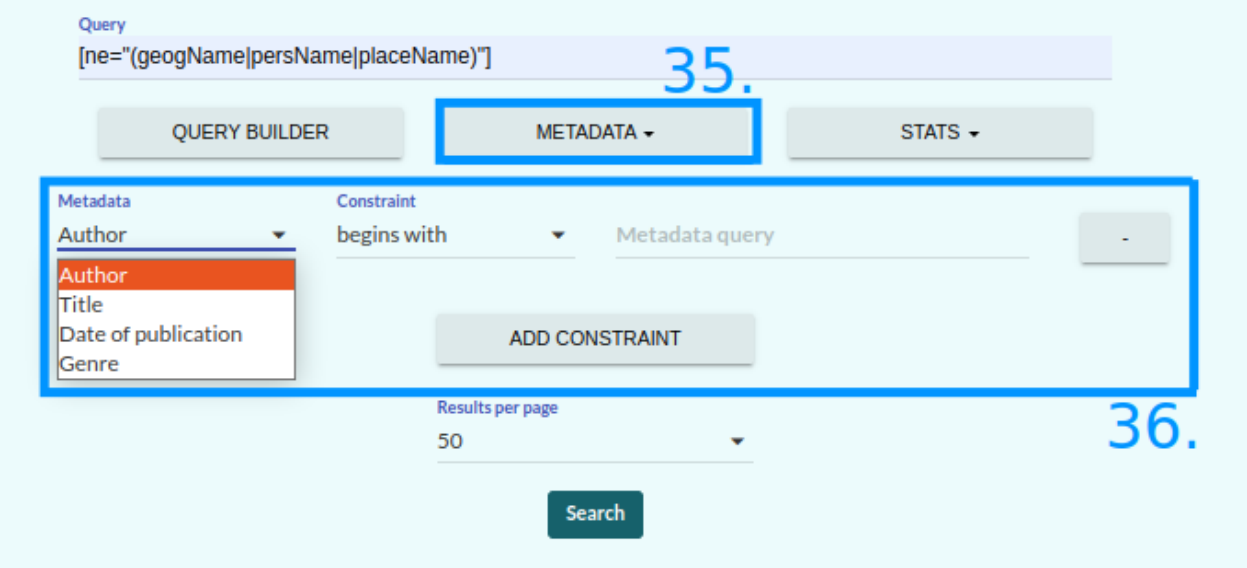
Write your query in the "Query" search bar (31), and press "Search" (34). The Corpus Query Language is described in detail in the following section. Press the button marked as (32) to use the visual query builder. Click on "Metadata" (33) to limit your search criteria using the metadata of your texts.



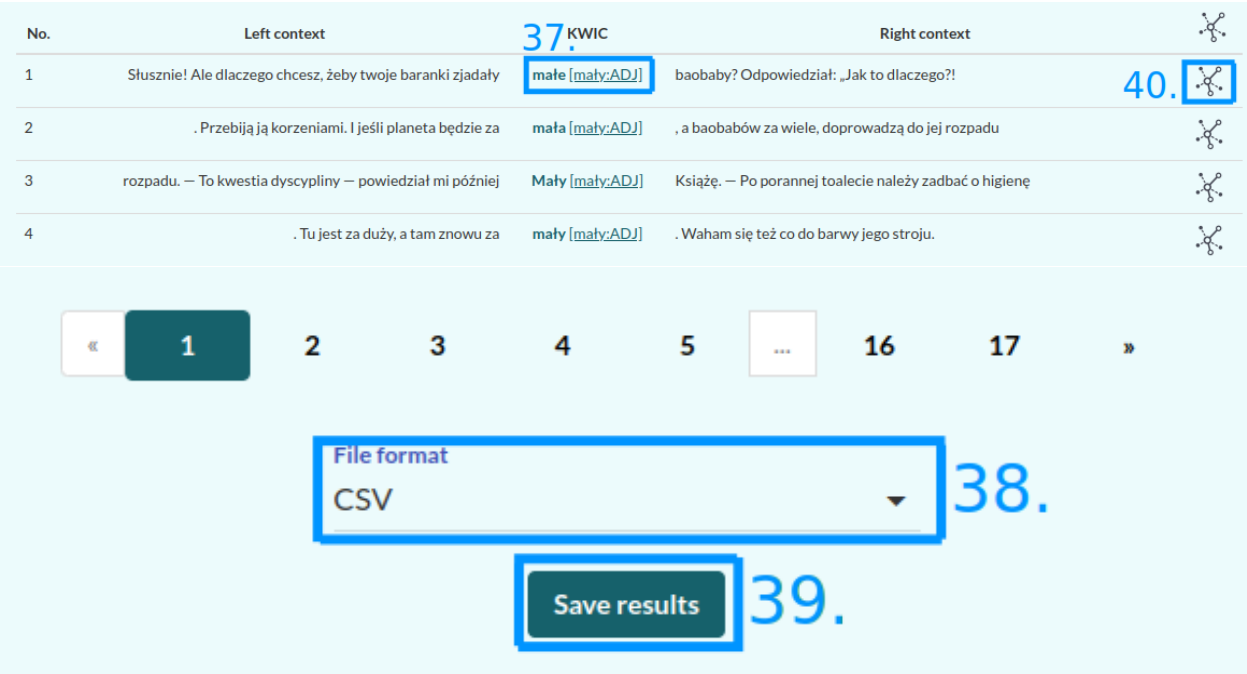
The visual query builder (32) allows you to form your query by selecting features of the items of interest from a drop-down list. Press "Save" to return to the search screen. Your query, "translated" into the Corpus Query Language, will appear in the search bar.



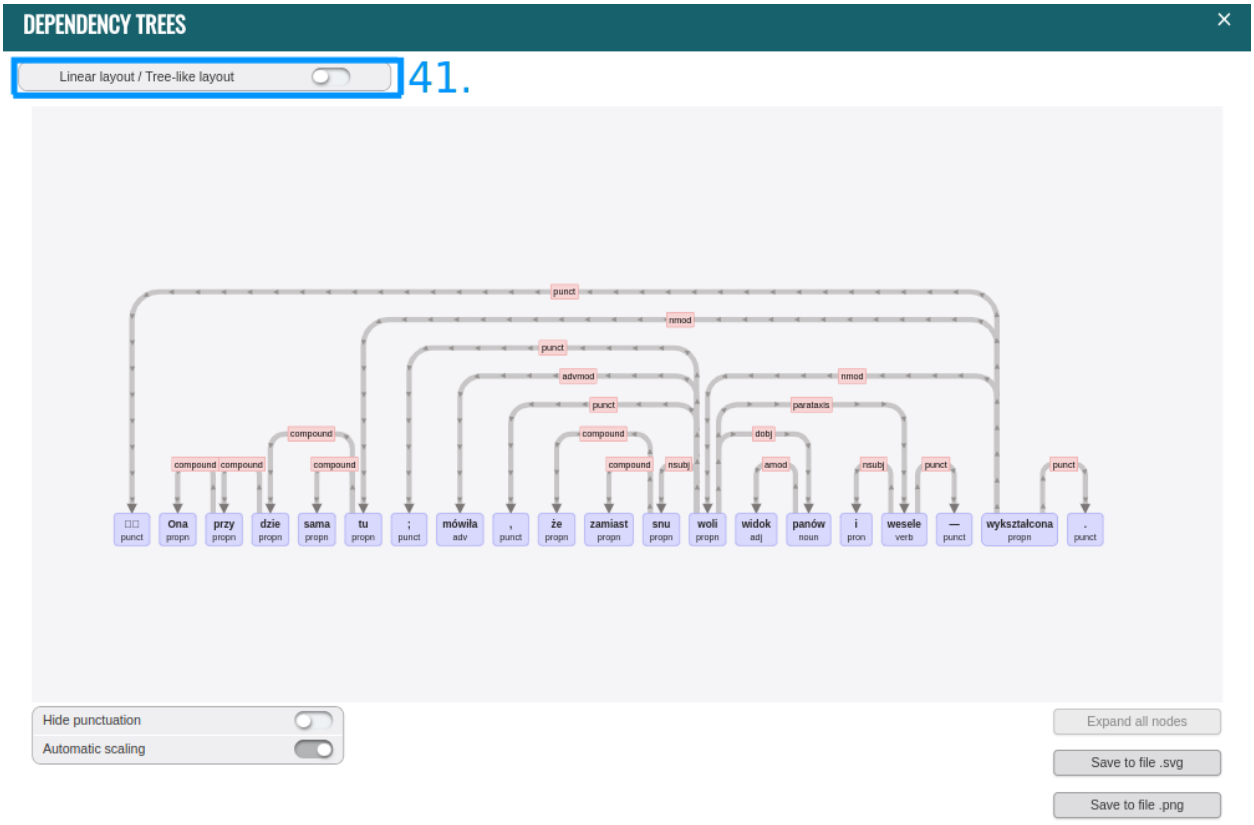
Press the button marked as (33) to drop down the Metadata menu (36). Now, you can limit your search to texts that fulfil some specific criteria.

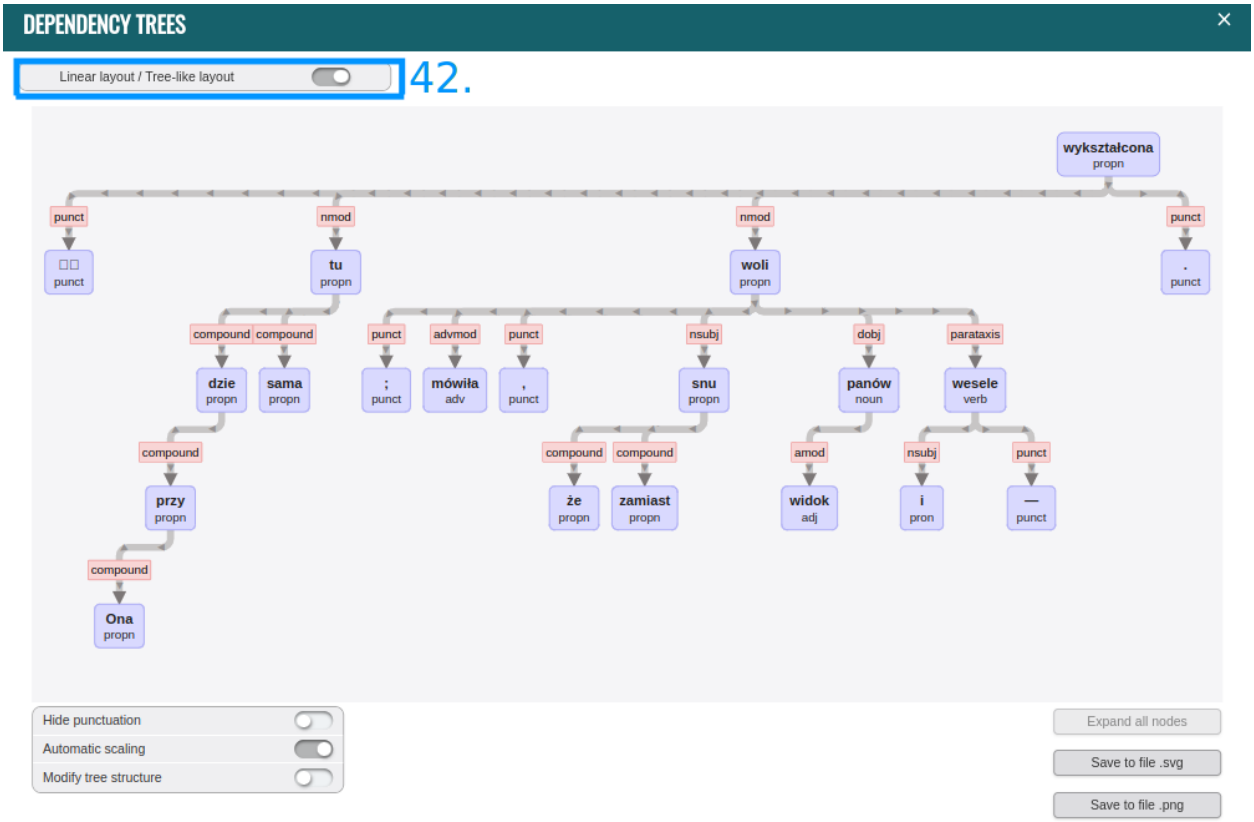


Once you press "Search" (34), the results of your query (in the form of a concordance) will be displayed. Clicking on the word or expression of interest (37) will display a wider context and the metadata of the text. In order to download the results of your query in CSV or XLS format, choose the file type and press "Save results" (39).



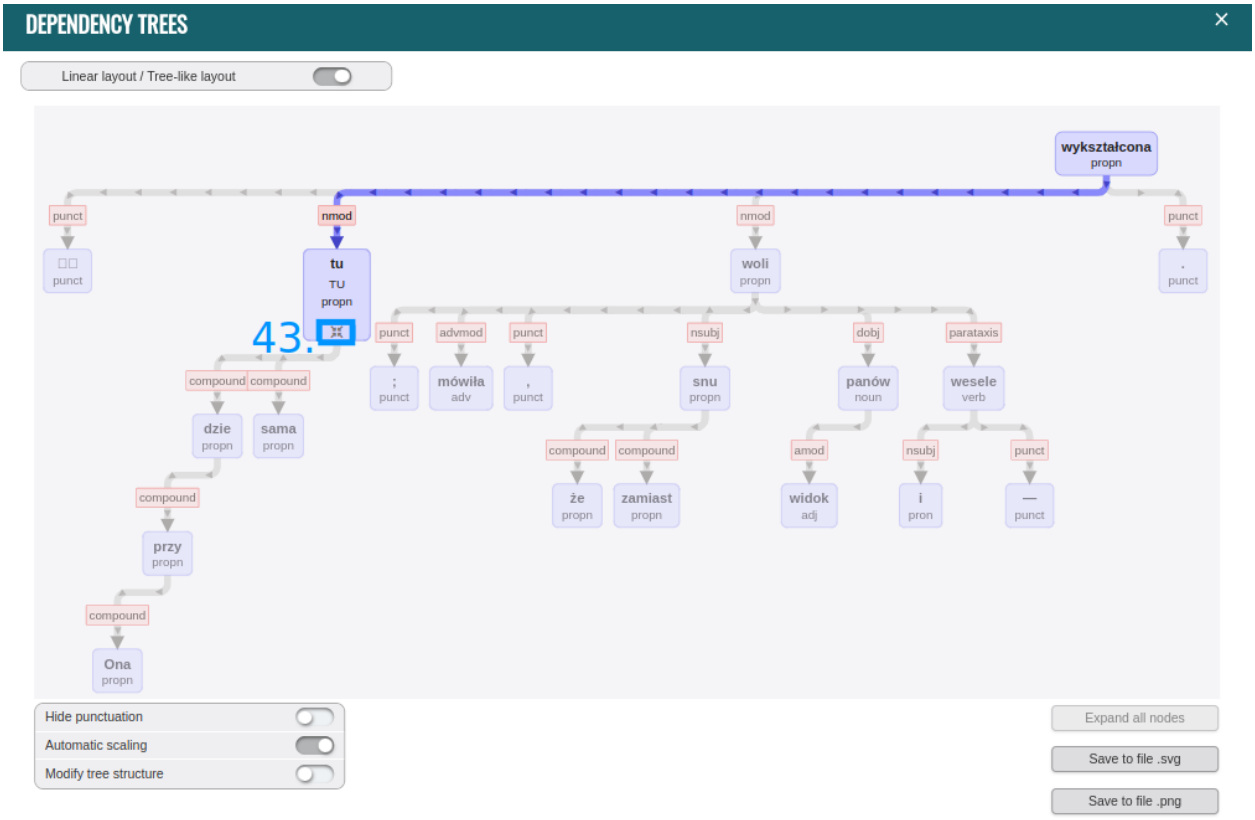
Clicking on the icon marked as (40) will allow you to examine visualisations of dependency trees. Specifically, the entire sentence, of which the word or expression of interest forms part, will be visualised. You can switch between two layouts: linear (41) and tree-like (42).





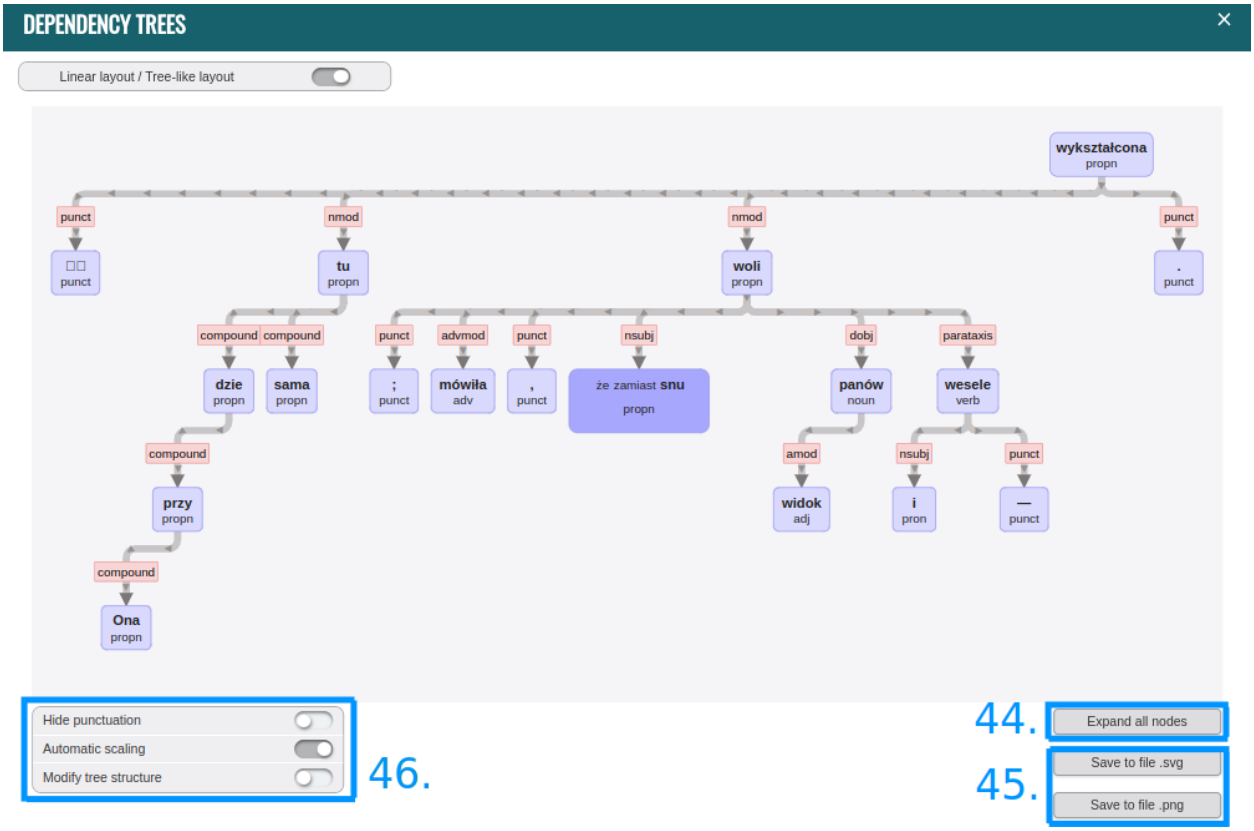
You can hide selected nodes by placing your cursor over the selected node and clicking on the icon marked as (43).

You can expand it in a similar manner. Or, you can press the “Expand all nodes” button (44).



You can save the visualisation in the SVG or PNG format (45). Additional options (46) include:

- hiding punctuation marks (this will make the tree easier to read),
- automatycznego pozycjonowania drzewa (po wybraniu tego przycisku analizowane drzewo zostanie usytuowane w centralnej części okna modalnego),
- modifying the structure of the tree (specifically, modifying the positioning of its nodes).

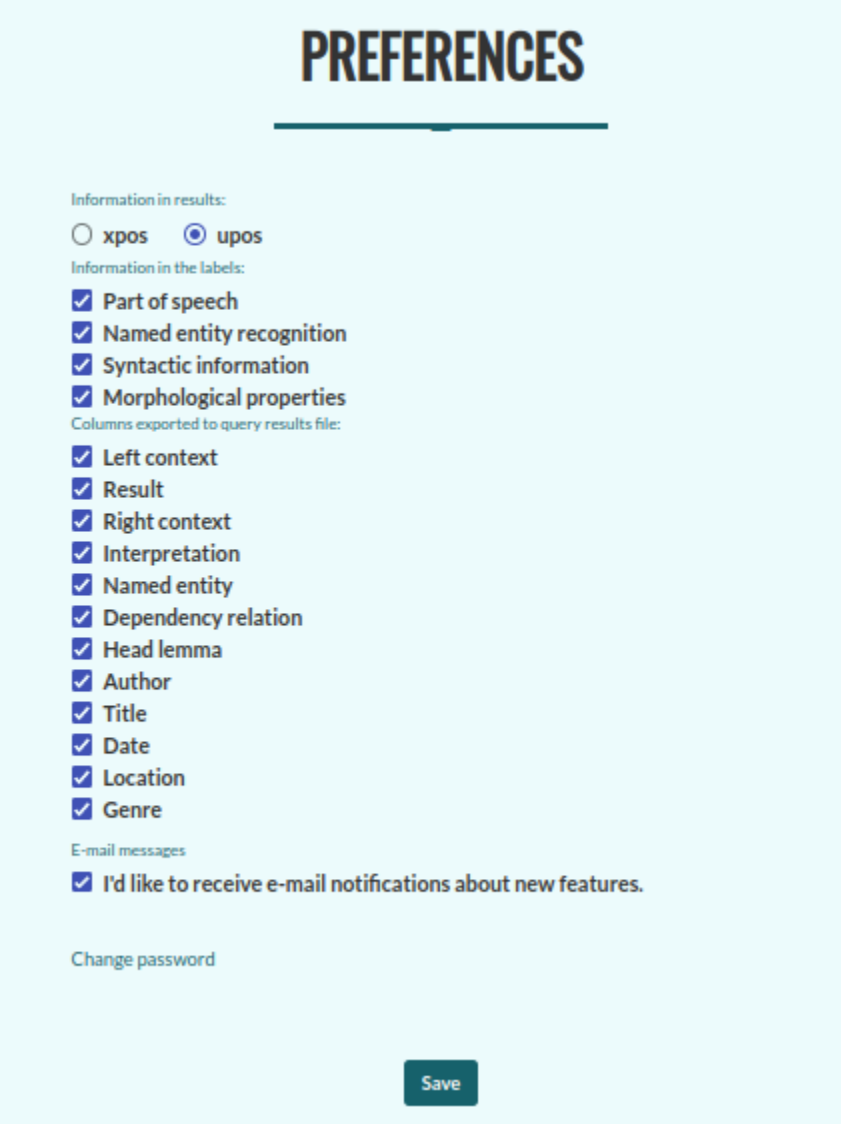


1.1.4 Preferences

From the main menu choose Preferences to change how the results of your queries are displayed. You can choose the tag (xpos or upos) that will be displayed in the main column of the results. You can also select layers of annotation. Information pertaining to the selected layers will be displayed when you place your cursor on an item that formed part of your search.

“Columns exported to query results file” allows you to select the pieces of information that will be included if you export the results of your query.

You can also change your password, and subscribe to the newsletter which will inform you about news on Korpusomat, or cancel your subscription.



1.2 Corpus Query Language

This section comprises a guide into the Corpus Query Language, which takesinto account the different layers of annotation available in Korpusomat.

1.2.1 Tokenisation

Morphosyntactic tags are attributed to tokens, which correspond roughly to words. A token cannot be longer than an orthographic word (separated by spaces from other orthographic words, punctuation marks excluded), but it may sometimes be shorter. Segmentation rules may be different for different languages, and they depend on the decisions made by the creators of language resources for a given language (mainly treebanks), and by the creators of specific programming tools. For instance, in Polish language corpora (including the Polish National Corpus), it is customary to separate past verb forms from their so-called agglutinates (markers of person and number), and from the marker of the conditional mood (*by* particle). As a result, one word may be divided into two or three component parts, and each of them is given its own morphosyntactic tag. For example, the Polish verb *jedlibyśmy* (*we would be eating*) would be separated into three different tokens: *[jedli][by][śmy]*, where *jedli* is the main verb (*eat*), *by* is the marker of the conditional mood, and *śmy* marks for the first person plural. In Korpusomat, the results of the tokenisation process may differ depending on the selected pipeline. Specifically, Stanza uses the same tokenisation model as is employed in the Polish National Corpus (it separates the agglutinate and the *by* particle), whereas spaCy considers past verb forms and conditional forms to constitute discrete tokens. This, however, is a rare and extreme case. Usually, the tokenisation of texts in a given language should be in line with the models adopted in the treebanks and the national corpora of this language, regardless of the pipeline employed.

1.2.2 Morphosyntactic tags

Wszystkie korpusy w Korpusomacie zawierają warstwę informacji morfosyntaktycznej zgodną ze specyfikacją Universal Dependencies. Informacja ta jest rozdzielona na dwie składowe: oznaczenie części mowy (tzw. UPOS — *universal part of speech*) oraz cechy morfosyntaktyczne (tzw. UFEATS — *universal features*). Obie te składowe (nazwy części mowy, nazwy cech morfoskładniowych i listy ich możliwych wartości) są opisane w dokumentacji na stronie projektu UD. Ponieważ z zasady jest to opis uniwersalny, każdy z konkretnych języków korzysta tylko z podzbioru cech morfologicznych i ich wartości.

Alongside morphosyntactic annotation based on the UD framework, an additional XPOS tag is available for most corpora. XPOS stores morphosyntactic information in line with the tagset employed in resources for a given language. Both pipelines return XPOS-tagged texts, but the precise forms of the tags depend on the decisions made by the creators of these specific tools and, in particular, by the creators of UD treebanks (specifically, technical descriptions of the tagset and the granularity of morphosyntactic descriptions may vary). As a result, XPOS tags do not have a single, standardised form. Usually, however, they comply with the tagsets employed in national corpora. If XPOS tags are not included in a given UD treebank (as is the case of Russian), Korpusomat will not be able to provide them (regardless of the pipeline employed). Sometimes, XPOS tags may differ depending on the pipeline, e.g. for Polish, using Stanza will return full morphosyntactic XPOS tags (as they are employed in Polish corpora), whereas spaCy will reduce them to the component which signals the grammatical class of the given word.

1.2.3 Corpus Query Language

The syntax of queries in MTAS in based on the Corpus Query Language (CQL), used in many other corpus browsers. This section describes the CQL as it is employed in Korpusomat.

MTAS is a multifunctional browser, ideal for corpora with many layers of annotation. The following is a guide to querying corpora indexed in Korpusomat, which are annotated at three different levels: morphosyntactic, syntactic, and the level of named entities. More general, basic information on MTAS is available at the project's website.

Searching for tokens

The most basic unit in a corpus search is a token (an orthographic word). Usually, each element of a query has to be enclosed in square brackets. You can specify that you are searching for a token by adding the attribute `orth`. In this way, you can search for one or more (neighbouring) tokens. For example, the following query will return all instances where the words "green" and "ideas" appear next to each other (but it will not return thos

```
[orth="green"][orth="ideas"]
```

You can also search for tokens by simply typing them into the search box:

```
green ideas
```

By default, the tool distinguishes between lowercase and uppercase letters. As a result, the following queries will return different results:

- `colorless`
- `Colorless`

You can, however, use the attribute `orth_lc` (where `lc` stands for *lower case*) to change all uppercase letters in a token into lowercase letters. As a result, a query such as `[orth_lc="colorless"]` will return all instances of words such as *colorless*, *Colorless*, *COLORLESS*, and *COlorLESs*.

Queries may also be based on standard regular expressions, which employ special characters such as `?`, `*`, `+`, `.`, `,`, `|`, `,`, `[`, `]`, `(`, `)`, as well as natural numbers (written in Arabic numerals), e.g. `0` or `21`. While a more formal description of regular expressions is beyond the scope of this guide, the following examples illustrate the most essential rules of their use.

1. ```
   [orth="(pal|gal)"]
   ```

   `|` is an equivalent to "or": it matches all instances of one of two tokens (which are placed inside round brackets), such as *gal* or *pal*,

2. ```
   [orth="[gp]al"]
   ```

   placing *p* and *g* inside square brackets, followed by *al*, will return all instances of words beginning with either *p* or *g*, followed by *al*, in the exact same manner as the query above,

3. ```
   [orth="gall?"]
   ```

   `?` makes the element it follows (or a group of elements placed inside round brackets) optional. The above query will return all instances of *gal* and *gall*,

4. ```
   [orth="gal."]
   ```

   full stop substitutes for one character only. This query will return all instances of tokens such as *gall*, *gale*, *gala*, *gals*, but not *gal* or *galled*,

5. ```
   [orth="gal.?"]
   ```

   gal, *gals*, *gale*, *gala*, *gall*, but not *galled*,

6. ```
   [orth="t.t.."]
   ```

   tokens with five characters, where *t* is the first and third character (e.g. *tutor*, *totem*, *total*),

7. `[orth="w*i"]`

   asterisk repeats the preceding character or a group of characters any number of times. For instance, we can use it to search for tokens consisting of the letter *w* followed by zero or more letters *i*, such as *w*, *wi*, *wii*, *wiii*, etc.,

8. `[orth="gal.*"]`

   tokens beginning with *gal*, such as *gal* or *gallery*,

9. `[orth=".*al+"]`

   the plus sign has a similar function: it repeats the preceding character or a group of characters any number of times higher than zero. The above query will return tokens which end in *al*, *all*, *alll*, etc., but not in *a*, for example *gal*, *regal*, or *gall*,

10. `[orth="wi{1,3}.*"]`

    w is followed by between 1 and 3 letters *i*, which are then followed by any number of characters. The results might include tokens such as *wi*, *wii*, and *winter*,

11. `[orth=".*(ha){3,}.*"]`

    `n,` repeats the preceding character or a group of characters (placed inside square brackets) at least n times. For example, the above query will return tokens where *ha* is repeated at least three times, such as *ahahaha* or *hahahaha*,

Queries with different attributes

In order to find all forms of the word *corpus*, type in the following query:

```
[lemma="corpus"]
```

`lemma` is one of the many attributes which may form part of a query. It takes the basic, dictionary form of a word as its value. For instance, `[lemma="sleep"]` will return forms such as *sleep*, *slept*, and *sleeping*.

Similarly as in the case of the attribute `orth`, regular expressions may be used to form queries about lemmas. For example, the following

```
[lemma="p[ao]tent"]
```

will return all instances of tokens whose dictionary forms (lemmas) are either patent or potent.

You can also specify more than one attribute of your search item. For instance, if you wish to find all instances of the token *cooler*, but only in its adjectival meaning (i.e. the comparative form of the adjective *cool*), but you want to exclude those instances where *cooler* is used as a noun, the following query may be employed:

```
[orth="cooler" & lemma="cool"]
```

The following query, which instead excludes those instances where *cooler* is used as a noun, will return similar results:

```
[orth="cooler" & !lemma="cooler"]
```

While `&` is the operator of logical conjunction, a disjunctive formula is notated with `|`. Here are some examples of how the latter can be employed:

- `[lemma="sleep" | lemma="dream"]`

  returns all forms of the verbs sleep and dream; this query is equivalent to `[lemma="sleep|dream"]`,

- `[lemma="sleep" | orth="dreaming" | orth="dreamer"]`

  returns all forms of the verb sleep, as well as all instances of the segments *dreaming* and *dreamer*,

- `[orth="cooler" & !(lemma="cool" | lemma="cooler")]`

  returns all instances of the token *cooler* which are neither forms of the lexeme cool, nor forms of the lexeme cooler.

In order to better understand the difference between the two operators - `&` and `|` - let us compare the following two queries:

```
[orth="cooler" & lemma="cool"]
[orth="cooler" | lemma="cool"]
```

The first query matches only tokens which are interpreted as forms of the lexeme cool. The second query matches all tokens of the word *cooler*, regardless of their interpretation (noun or adjective), as well as all forms of the lexeme cool, such as, among others *cooling*, *coolest*, and *coolers*.

A query may include as many attributes (with their values) as necessary, which may be connected using operators such as `!`, `&`, and `|`, as the examples above demonstrate. But a query without any conditions is also possible. The following query may be used to retrieve all tokens in the corpus.

```
[]
```

In other words, empty square brackets stand for any token. They can be used, for instance, to retrieve two specific tokens, which are separated by two unspecified tokens:

```
[orth="you"][][][lemma="sleep"]
```

This will retrieve multi-word items such as *you can always sleep* or *you and I sleep*.

You can also search for two tokens which are separated by up to n unspecified tokens. For example, the following query will return multi-word expressions, where *you* is separated from *sleep* by two, three, or four unspecified tokens:

```
[orth="you"][]{2,4}[lemma="sleep"]
```

The results will include expressions retrieved using the preceding query (*you can always sleep* and *you and I sleep*), as well as others, such as *you can go to sleep* or *you and I will sleep*.

Queries using morphosyntactic tags

The attribute `upos` (*universal part of speech*) takes as its values different grammatical classes; their symbols are listed here: https://universaldependencies.org/u/pos/index.html. You can use it, for example, to search for two adjacent nouns, both beginning with *a*:

```
[upos="NOUN" & orth="a.*"]{2}
```

Similarly as in the case of orthographic words and lemmas, you can also use regular expressions in queries about grammatical classes.

Furthermore, using the `xpos` attribute, words marked with tags specific for the given language may be retrieved. Here, too, regular expressions may be employed. For example, you can search for all singular neuter nouns in the nominative case in a corpus of Czech using the following query:

```
[xpos="NNNS1.*"]
```

In order to retrieve nouns of the same characteristics in a Polish corpus (using Stanza) you would need to type in the following query:

```
[xpos="subst:sg:nom:n.*"]
```

In both cases, the values of the `xpos` attribute are followed by `.*` to reflect the fact that each tagset may also include other categories, apart from those that are specified in each query (part of speech, number, gender, and case).

Thus, you can search for orthographic forms (attribute `orth`), lemmas (attribute `lemma`), and tokens belonging to specific grammatical classes (`upos` or, alternatively, `xpos`). You can also specify the values of different grammatical categories - such as gender or case - provided that they are included in the grammar of the language under the investigation. The layer of morphosyntactic features (UFEATS) of a given language treebank includes all of the categories that can be applied to the words of this language; their lists are available at <https://universaldependencies.org/u/feat/all.html>`__.

For instance, if the grammar of the language of the corpus includes the property of grammatical number, you can form the following queries:

1. ```
   [number="sing"]
   ```

   matches all singular forms,

2. ```
   [upos="NOUN" & number="sing"]
   ```

   matches singular common nouns,

3. ```
   [upos="NOUN" & !gender="fem"]
   ```

   matches common nouns with the value of gender other than feminine (e.g. masculine and neuter for languages such as Polish, Czech, or Ukrainian),

4. ```
   [number="sing" & case="(nom|acc)" & gender="masc"]
   ```

   matches singular nominative or accusative masculine forms (if the grammar of the given language has the categories of number, case, and gender).

The names of the categories can also be substituted by the universal attribute `ufeat`. For example, the following two queries will return the same results:

```
[upos="NOUN" & case="acc" & number="plur" & gender="fem"]
```

```
[upos="NOUN" & ufeat="acc" & ufeat="plur" & ufeat="fem"]
```

Visual CQL query builder

The visual CQL query builder may be employed to create simple queries. You can use it to define the attributes of each element of a query: part of speech, lemma, as well as values of all grammatical categories listed on the UD website: https://universaldependencies.org/u/feat/all.html. Different conditions may be included using the operators *and* (conjunction) and *or* (disjunction). Once each element of a query is defined, press the *Save* button. The query will now appear in the search bar, so that you can verify whether it is correct. Before pressing the *Search* button, you can define additional search parameters (this is optional): for instance, you can use metadata to limit your search.

Limiting the search to a sentence or a paragraph

In corpora indexed in Korpusomat, the units of organisation are sentences and paragraphs. You can use this division when building your queries, for example by limiting a query to a single sentence.

In order to limit a query, you need to add the keyword `within`, followed by <s/> or <p/>, depending on whether you want to limit it to a single sentence (<s/>) or paragraph (<p/>). For example, the following query matches sentences where the lemma sleep is separated from the token *furiously* by at least one, but no more than ten segments:

```
[lemma="sleep"][!orth="furiously"]{1,10}[orth="furiously"] within <s/>
```

Dodatkowo można również na elementy <s/> i <p/> nałożyć pewne warunki dotyczące tego, czy zawierają segmenty innego typu. Przykładowo, za pomocą następującego zapytania można znaleźć wszystkie wystąpienia czasownika pomocniczego być w czasie przyszłym ograniczone do zdań zawierających formę bezokolicznika:

```
[upos="AUX" & lemma="być" & tense="fut"] within (<s/> containing [verbform="inf"])
```

Intencją takiego zapytania jest odnalezienie (w przybliżeniu) wszystkich wystąpień konstrukcji czasu przyszłego złożonego, w których pojawia się bezokolicznik. Wśród wyników będą oczywiście również takie zdania, w których czas przyszły został utworzony z użyciem formy przeszłej czasownika, a bezokolicznik pełni w zdaniu inną funkcję gramatyczną. Można też sformułować zapytanie odwrotnie i zapytać o zdania, w których forma przeszła w ogóle nie występuje:

```
[upos="AUX" & lemma="być" & tense="fut"] within (<s/> !containing [tense="past"])
```

The full list of keywords which can be employed in an MTAS search is available here: <https://meertensinstituut.github.io/mtas/search_cql.html>`__. It is important to note, however, that not all of them will have their uses in Korpusomat.

Apart from tags marking elements of a text's structure, such as <s/>, there are also tags marking their beginnings and ends, such as <s> and </s>, respectively. They will not retrieve any specific segments, but they may be used to further limit the search for an already specified segment. For instance, the query:

```
<s> [upos="NUM"]
```

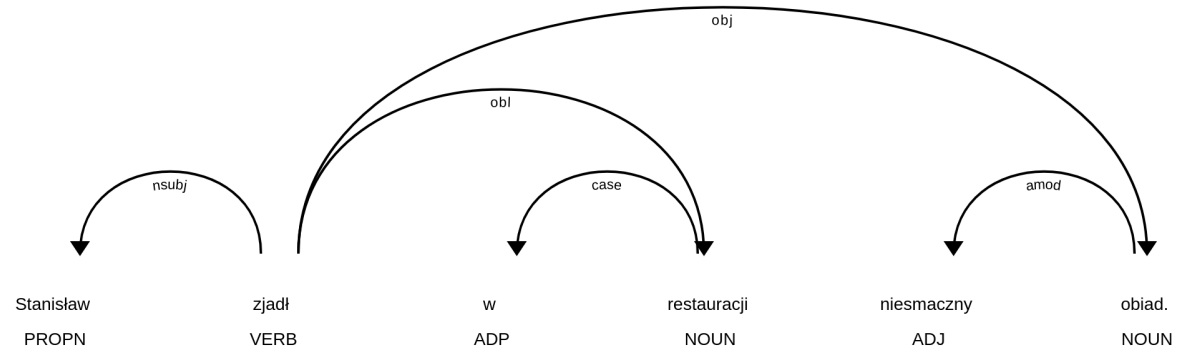
will retrieve all instances of a numeral at the beginning of a sentence. A similar query:

```
[upos="NUM"][upos="PUNCT"]</s>
```

will retrieve all instances of a numeral followed by a punctuation mark at the end of a sentence.

The layer of syntax

Another layer of annotation is based on dependency parsing. A text uploaded by a user is automatically divided into sentences, which are then analysed syntactically, in line with the principles of the Universal Dependencies project. The following image shows an example.



As MTAS is not designed to search for syntactic structures, it cannot be employed to index or search for full sentence diagrams. However, at the level of a token, Korpusomat indexes information about its syntactic head (specifically, the head's lemma and inflectional class), and about the type of relation between these two elements. It also indexes their relative positions (in the linear order) and the distance between them (measured in tokens). As a result, one can search for simple syntactic structures as well as analytical discontinuous inflectional forms.

The following attributes are available at the level of syntactic annotation:

- `deprel` — type of relation between a token and its syntactic head; its value may be one of the 65 dependency relations listed at <https://universaldependencies.org/u/dep/index.html>`__ (although the sets of values may differ between languages),
- `head.upos` — part of speech (UPOS) of the head of the given token,
- `head.lemma` — lemma of the head of the given token,
- `head.ufeat` — value of any morphological feature of the head of the given token,
- `head.distance` — distance from the token to its syntactic head,
- `head.position` — position (left- or right-branching) of the head with respect to the given token in a linear order.

As a result of adding these attributes to the Corpus Query Language, the user can easily find, for instance, all common nouns employed in the function of a direct object of a given verb:

```
[upos="NOUN" & deprel="obj" & head.lemma="dream"]
```

Or, alternatively, one can search for the verbs which take a specified noun as their direct object:

```
[deprel="obj" & head.upos="VERB" & lemma="idea"]
```

Należy jednak zwrócić uwagę, że w powyższym przykładzie wynikiem zapytania będą wystąpienia rzeczownika osoba, nadrzędne względem nich formy czasownikowe (finitywne i niefinitywne) będą się zaś znajdowały w lewym lub prawym kontekście wyników wyróżnione pismem pogrubionym. Można je jednak zgrupować i posortować względem ich częstości dzięki opcjom Statystyk.

The attribute coding for left- and right-branching of the syntactic head with respect to the given token allows the user to find examples of non-standard word order. For example, the following queries will retrieve instances when the verb is followed by its subject:

```
[deprel="nsubj" & head.position="left"]
```

or when the verb is preceded by its direct object:

```
[deprel="obj" & head.position="right"]
```

Similarly, a query which does not specify the position of the head, as in:

```
[upos="ADJ" & deprel="amod" & head.lemma="sleep"]
```

will match all adverbs modifying the verb sleep. In turn, specifying its position will limit the search to adverbs positioned to the left (*furiously sleep*) or to the right (*sleep furiously*) of the verb.

Thanks to the partial syntactic annotation, the user can search for elements of a phrase which remain in a relation of dependency, regardless of the fact that they may not necessarily be neighbouring words, but may, in fact, be separated by other segments. The attribute of distance further limits the search to instances where these elements are not neighbouring words:

```
[deprel="obj" & head.upos="VERB" & tense="past" & !head.distance="1"]
```

The above query will retrieve all words in the function of a direct object of a verb in the past tense, separated from the verb by at least one element.

Syntactic annotation may also be employed to search for passive forms of verbs:

```
[upos="AUX" & deprel="aux:pass" & head.upos="ADJ"]
```

którego dopasowaniem są słowa posiłkowe konstrukcji biernej połączone z formą imiesłowu biernego (oznaczoną jako przymiotnik) relacją `aux:pass`.

The layer of named entities

The final layer of information added to texts indexed in Korpusomat is the layer of named entities. Named entities are one- or multi-word items designating people, places, institutions, etc. There is no single, universal standard for annotating named entities, nor is there a multilingual dataset annotated consistently for named entities. As a result, the values and their ranges at this layer of annotation vary depending on the pipeline as well as across languages within each pipeline. Furthermore, for some languages, models of annotation of named entities are not available.

The most basic commonly used set of labels for named entities consists of only four elements: `PER` (person), `LOC` (location), `ORG` (organisation), and `MISC` (miscellanea). It is available, within the Stanza pipeline, for Spanish, French, Russian, and Ukrainian, among others. However, some languages employ more detailed classifications. For instance, there are 18 values for the classification of named entities in English and Chinese. The visual CQL query builder provides the full list of values available for a given corpus. The following examples employ the simplest four-element classification.

Named entities, similarly to sentences and paragraphs, may cross segment boundaries. Therefore, the `<ne />` tag may be used to refer to items marked as named entities. The rules regarding the use of slash also apply here:

- `<ne>` marks the beginning of a string marked as a named entity,
- `</ne>` marks the end of a string marked as a named entity.

The following is the simplest query about named entities:

```
<ne />
```

It will retrieve all named entities of every type in the corpus. But the user can also limit the search to, for example, names of locations:

```
<ne="LOC" />
```

Similarly as in the case of sentences and paragraphs, it is possible to create queries about specific orthographic or morphosyntactic features of named entities. For example, the user may search for:

```
[upos="CCONJ"] within <ne="PER" />
```

names of organisations where segments are linked by a coordinate conjunction, such as *Department for Culture, Media, and Sport* or *Organisation for Economic Cooperation and Development*,

```
<ne="LOC" /> [upos="CCONJ"] <ne="LOC" />
```

— wystąpienia dwóch nazw geograficznych połączonych spójnikiem współrzędnym, np. *Europa Zachodnia lub Skandynawia*,

```
[orth="A.*"][orth="M.*"] fullyalignedwith <ne="PER" />
```

two neighbouring tokens starting with, for instance, *T* and *M*, and forming a person's name, e.g. *Theresa May*, *Thomas Mann*.

Furthermore, information about the length of each named entity (measured in tokens) is available. The following query will return all named entities which consist of exactly three tokens:

```
<ne.len="3" />
```

odnajdzie wszystkie takie jednostki składające się z dokładnie trzech segmentów.

Using metadata to limit the search

Texts uploaded into Korpusomat are, by default, given four metadata labels: author, title, date of publication, and genre. This information is usually provided by the user, but the labels may also remain empty. The user may also add new labels.

The metadata may be used to limit corpus searches. In order to do so, press the Metadata button, define a constraint, and then press Add constraint. This process can be repeated in order to add more than one constraint.

1.3 Word profiles

1.3.1 Introduction

Profile słów umożliwiają odnalezienie w tekście słownictwa, które często łączy się ze wskazanym słowem w związki składniowe określonego rodzaju. Na przykład rzeczownik *oczy* często jest modyfikowany przez przymiotnik *niebieskie* i często jest dopełnieniem bliższym czasownika *zamknąć*. Z kolei rzeczownik *pies* często pojawia się w koordynacji z rzeczownikiem *kot*. Otrzymane kolokacje charakteryzują język korpusu, tj. w korpusie reprezentatywnym dla standardowego języka polskiego będą się pojawiały głównie związki wynikające z ogólnych zależności semantycznych lub frazeologii, natomiast w korpusie dziedzinowym — związki wywodzące się z języka danej dziedziny, związki charakteryzujące styl autora lub jego sposób myślenia. Na przykład w korpusie ogólnym słowo *funkcja* będzie często określane przymiotnikiem *podstawowa*, zaś w korpusie matematycznym częściej pojawi się przymiotnik *ciągła* lub

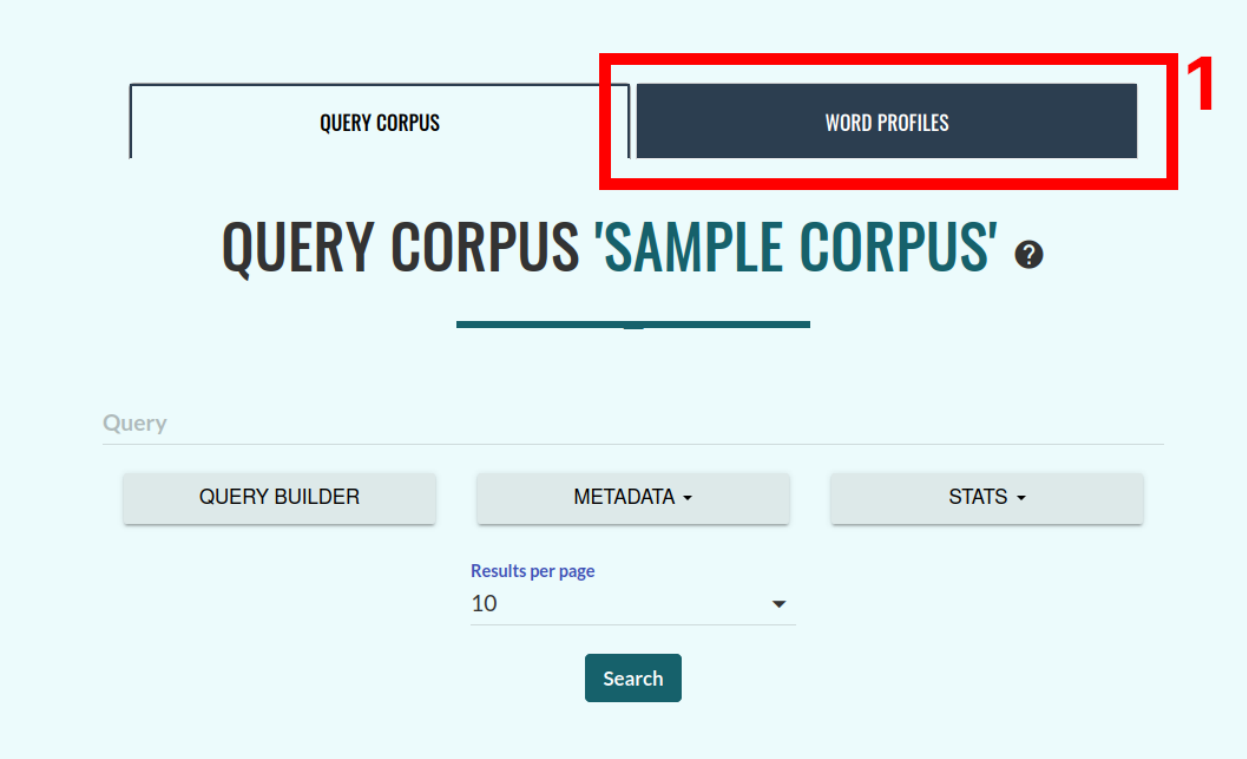
różnowartościowa. Można się też spodziewać, że przymiotnik *robotniczy* będzie występował z innymi kolokatami w korpusie z czasów PRL, a z innymi w korpusie współczesnym.

Note: Word profiles are only available for corpora which have the dependency annotation enabled.

Please note that, due to statistical reasons, the search quality is higher for large corpora (over 1 million tokens) and for relatively frequent words.

The process of calculating a word profile might take up to a few dozens seconds, depending on the size of the corpus and the frequency of the search word.

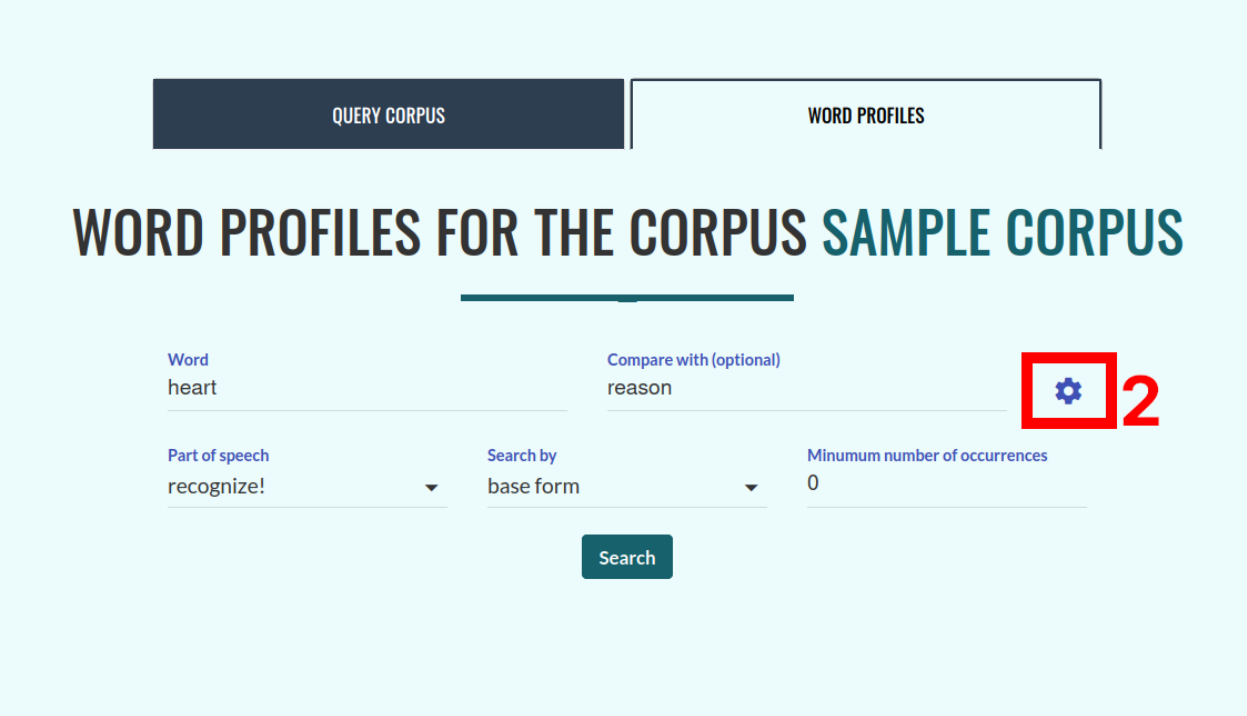
1.3.2 Usage



Profile słów są dostępne z poziomu ekranu *Odpytaj korpus*, karta *Profile słów* (**1** na obrazku). W pole *Słowo* należy wpisać słowo, którego profil chcemy wyliczyć.

Advanced search options

Po kliknięciu w ikonę koła zębatego (**2**) dostępne są również zaawansowane opcje wyszukiwania. Tworząc profil danego słowa, możemy wybrać, czy interesują nas wszystkie jego wystąpienia, niezależnie od formy w tekście (i.e. szukamy wg lematu), czy też chcemy zobaczyć jedynie kolokaty określonej formy danego leksemu (np. rzeczownika *psy*, a więc słowa w liczbie mnogiej, i mianowniku lub bierniku). Możemy też odfiltrować kolokaty, ustawiając minimalną liczbę wspólnych wystąpień w korpusie, ta funkcja pozwala ominąć pary, które rzadko się powtarzają, a uzyskały wysoki wynik ze względu na rzadkość ich poszczególnych elementów składowych w korpusie.



Formularz pozwalający doprecyzować parametry profilu słów: narzucić określoną interpretację pod względem klasy gramatycznej, określić, czy interesują nas wystąpienia wskazanej formy czy wszystkich form przynależących do danego leksemu, zastosować filtrowanie frekwencyjne lub słowo kontrastowe.

The results of the search are displayed in a table where each column represents one of the syntactic relationships characteristic of the word. The collocates in each column are arranged according to the strenght of the association

(descending), independently of the rankings for the other syntactic relationships.

Note: The default value of the minimum number of occcurrences of each of the collocates depends on the size of the corpus – for corpora with at least 100 thousand tokens this number equals 3, while for other corpora it is 0.

Comparative profiles

Aplikacja umożliwia także tworzenie profili porównawczych. W tym celu należy wpisać do pola *porównaj z* drugie z interesujących nas słów. Wyszukiwanie porównawcze zakłada, że zadane słowa należą do tej samej klasy gramatycznej. Przygotowując tabelę, aplikacja weźmie pod uwagę różnicę wartości **logDice** słowa podstawowego, oraz słowa porównawczego dla każdego z kolokatów. Tabela jest automatycznie skracana do postaci w której ekstrahowane są trzy sekcje. Kolokaty wyraźnie preferujące pierwsze słowo, kolokaty neutralne (o wartościach różnicy logDice najbliższych 0) oraz kolokaty wyraźnie preferujące słowo porównawcze. Indeksy wierszy wpadających do każdej z tych sekcji są oznaczone innym kolorem.

COMPARATIVE SEARCH RESULTS FOR WORDS SPIRIT AND REASON AS A/AN COMMON NOUN (NOUN)
THESE WORDS APPEAR IN THE CORPUS 90 AND 320 TIMES, RESPECTIVELY

Visible columns

Search:

	words which have "spirit" vs. "reason" as nominal subject	words which have "spirit" vs. "reason" as direct object	words for which "spirit" vs. "reason" is a modifier	appositions of the word "spirit" vs. "reason"	words which form coordination with "spirit" vs. "reason"	adjectival modifiers of the word "spirit" vs. "reason"	numeric modifiers of the word "spirit" vs. "reason"	clausal modifiers of the word "spirit" vs. "reason"	nominal modifiers of the word "spirit" vs. "reason"	words which have "spirit" vs. "reason" as their nominal modifier	determiners of the word "spirit" vs. "reason"	adpositions of the word "spirit" vs. "reason"	words with which "spirit" vs. "reason" forms multiword expression
1	temper VERB 8.461	weaken VERB 8.445	talk VERB 7.978	man NOUN 4.777	word NOUN 6.275	right ADJ 6.84		meadow NOUN 8.476	writings NOUN 9.497	variance NOUN 8.313			
2	tarry VERB 8.492	waste VERB 8.3	propose VERB 8.057	horse NOUN 8.093	view NOUN 7.371	philosophic ADJ 8.445		increase VERB 8.142	wealth NOUN 7.978	type NOUN 8.272		like ADP 6.057	
3	see VERB 5.884	unite VERB 7.212	poor ADJ 8.046	dog NOUN 8.0	unity NOUN 8.081	own ADJ 5.975		fight VERB 8.105	tyrant NOUN 7.743	quality NOUN 7.453		into ADP 5.606	question NOUN 6.776
4	appear VERB 0.851	teach VERB 1.428	be VERB 0.161	animal NOUN 7.212	do VERB 0.405	other ADJ 0.237	two NUM -5.58	be VERB 0.161	that PRON -0.788	element NOUN 0.758	a DET 0.559	in ADP 1.795	soul NOUN 0.491
5	have VERB -0.828	have VERB 0.364	give VERB -0.77	be VERB -3.912	what PRON 0.292	like ADJ 0.166		he PRON 0.049	they PRON -1.258	one NOUN 0.541	all DET 0.218	with ADP -0.232	man NOUN -4.523
6	be VERB -2.793	make VERB -0.661	say VERB -0.872	lie VERB -6.504	philosophy NOUN -0.133	same ADJ -0.312		say VERB -4.596	he PRON -1.272	any DET -4.539	the DET -0.932	without ADP -1.671	
7	assert VERB -6.595	comprehend VERB -6.578	accompany VERB -6.599	friend NOUN -5.616	cause NOUN -6.186	epic ADJ -6.665		ashamed ADJ -6.612	affection NOUN -7.985	assistance NOUN -6.625	what DET -6.069	for ADP -9.889	
8	arise VERB -7.905	care VERB -6.553	accept VERB -6.484	Thrasymachus PROPN -6.415	call VERB -5.754	chief ADJ -6.591		appear VERB -5.989	State PROPN -6.449	apprehend VERB -6.582	that DET -6.667	by ADP -4.7	
9	allow VERB -6.139	assist VERB -6.616	abhor VERB -6.656	APOLLODORUS PROPN -6.629	be VERB -4.912	analogous ADJ -6.66		affirm VERB -7.445	God PROPN -5.961	admonition NOUN -6.656	another DET -6.264	between ADP -6.006	

Items from 1 to 9 (total of 9)

Comparative search results: *spirit* vs *reason* in Plato's dialogues.

Clicking any of the collocates will generate a query that will allow to find all common occcurrences of the two terms.

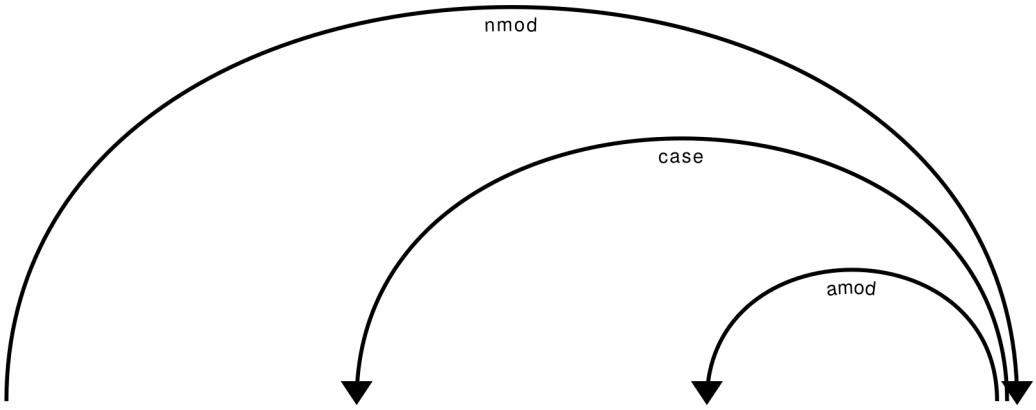
1.3.3 Metrics used

Profile słów przedstawiają słownictwo często współwystępujące ze wskazanym słowem. Znaczenie słowa *często* jest tutaj formalizowane za pomocą miary **logDice** (i to te wartości są widoczne w tabeli). Miara ta przypisuje każdej z badanych par słów wynik będący – w pewnym uproszczeniu – stosunkiem liczby wystąpień w korpusie razem do sumy wystąpień w korpusie w ogóle (razem lub osobno) każdego ze słów. W ten sposób odfiltrowujemy takie słowa, które pojawiają się obok zadanego słowa często w wyniku tego, że same są bardzo częste (np. czasownik *mieć*, w odróżnieniu od czasownika *zamykać*).

The **logDice** is an interpretable measure used in collocate extraction. Its maximum value is 14 (when words always co-occur), and the difference between two values equal to 1 indicates that one of the collocations is twice as frequent as the other. The logDice value does not depend on the size of the corpus; it is thus possible to compare values calculated for different corpora.

1.3.4 Linguistic basis

Word profiles are calculated based on morphological annotation and dependency parsing results; hence this function is only available for corpora which have the dependency annotation layer. For each part of speech, there is a set of rules regarding potential collocates of the given word. For example, for nouns, the algorithm will search for verbs which take the given noun as their subject (*a person believes*) or their direct object (*to protect a person*), as well as nouns modified by the given noun (*a person of commitment*). By default, the set of rules is selected by the morphosyntactic class of the query word, recognised automatically by the application. However it is possible to impose a grammatical interpretation (e.g. *walk* as a noun, not as a verb). In the output table the collocates are displayed as lemmas.

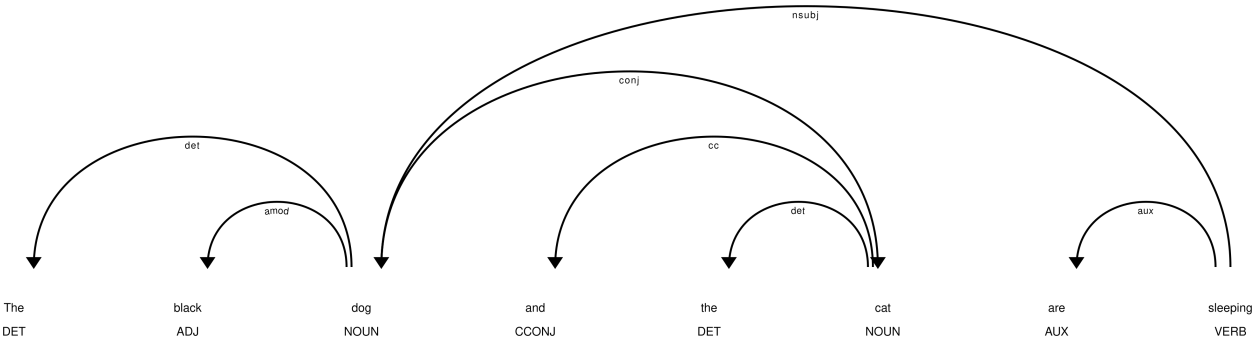


To calculate word profiles, dependency trees, such as the one shown in the picture, are used. In case of the word *heart* in the above example, *little* is an adjectival modifier of the word *heart*, and *person* is a word which has *heart* as its nominal modifier"

It should be noted that the search does not take negation into account. The occurences of a word will count towards a collocation regardless of whether they are within the scope of negating modifiers (e.g. *not*), conjuncts such as *nor*, lexical modifiers which are semantically close to negation (e.g. *little* in *little influence*), or are themselves formed via negation (e.g. *unseen*).

Coordination relationships

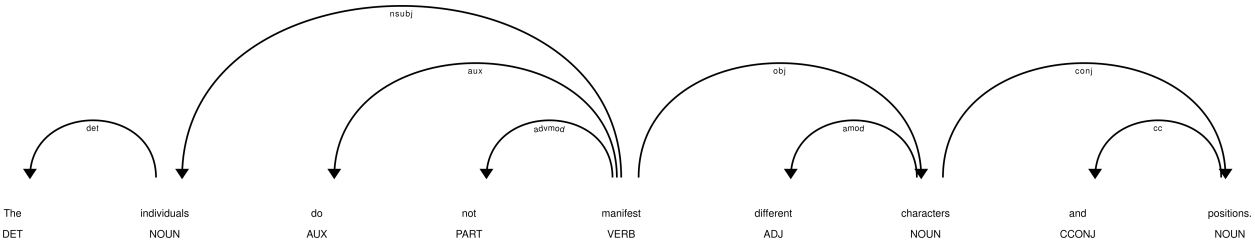
Coordinating conjunction is one of the most important relations. In the adopted grammatical formalism, coordination is represented as a subtree, in which the first component of the coordination is the head, and the subsequent units – the leaves. The head of the subtree is linked with its superordinate by the relation it would have if there were no conjuncts. Conjuncts, if there are any, are linked with the leaves by the *cc* relation. For example, in the sentence "The black dog and the cat are sleeping." the word *dog* is connected with the verb *sleep* by the *nsubj* relation, the word *cat* is a subordinate of the word *dog* linked to it by the *conj* relation, and *and* is related to *cat* via a coordinating conjunction, with the *cc* tag, as in the example below.



In order to include words which are the second or subsequent coordination components, in the collocate extraction system used coordination subtrees are handled in a special way. During the tree search the algorithm leaps over the root of the subtree, i.e. the first component of the coordination. Specifically, for the above example, both *dog* and *cat* will be recognised as subjects of the word *sleep*. Conversely, *sleep* will be recognized as a verb which has *cat* as its subject, even though these words are indirectly linked through the word *dog*.

This mechanism only applies to some of the possible relations. In the above example the word "black" will not be considered an adjectival modifier of the word "cat". The relations that trigger the mechanism of coordination extension include:

- for nouns (NOUN/PROPN):
  - *words which have (…) as their nominal subject*
  - *words which have (…) as their direct object*
  - *words which have (…) as their indirect object*
- for verbs (VERB):
  - *words which are the nominal subject of (…)*
  - words which are the direct object of (…)*
  - *words which are an indirect object of (…)*
  - *words which are the clausal subject of (…)*
  - *words which have (…) as their clausal subject*
  - *words which are a clausal modifier of (…)*
  - *słowa których dopełnieniem zdaniowym jest (…).*



The representation of coordination and negation relationships in the syntactic formalism used. As the first coordination component is the direct subordinate of the *obj* relationship, the word *character* will be recognised as a *direct object of "manifest"*. Moreover, during the calculation the algorithm moves along the *conj* edges to additionally include the word *position*.

1.4 Corpus statistics

1.4.1 Frequency list

This option displays the frequency list of all lemmas in the corpus. By default, the list is limited to content words, but there is an option to include or exclude any part of speech by clicking on the button at the top of the page. Clicking on the "Download" button allows the user to download the full frequency list without the applied filters.

Measures of dispersion and adjusted frequency

Apart from the frequency list, adjusted word frequencies and dispersion measures can also be consulted.

The available measures are listed below. The following notation conventions have been adopted in the formulas below:

- N – the number of texts in the corpus.
- N_t - the number of texts containing a given word.
- L – the number of words in the corpus.
- L_i – the number of words in the i-th text of the corpus.
- F: – the number of occurrences of the word in the whole corpus.
- F_i – the number of occurrences of the word in the i-th text of the corpus.
- $\bar{F}$ – the average number of occurrences of the word in a text: $\bar{F} = \frac{\sum_{i=1}^{N} F_i}{N}$.
- P_i – the ratio of the number of occurrences of the word in the i-th text of the corpus and the total number of words in the i-th text of the corpus.
- $\bar{P}$ – the average sation of the number of occurrences of the word and the total number of words in a corpus text: $\bar{P} = \frac{\sum_{i=1}^{N} P_i}{N}$.
- S_i – the ratio of the number of words in the i-th text of the corpus nad the total number of words in the corpus: $S_i = \frac{L_i}{L}$

Measures of dispersion and adjusted frequency:

- Frequency – the number of occurrences of a word in the corpus.

$$F$$

- IDF – a measure of the amount of information related to the use of a given word. The value of IDF is equal to 0, if the word is present in all of the texts, and the lower the number of texts containing the word, the higher the value.

$$log_2 \frac{N}{N_t}$$

- Variation coefficient (*vc*).

$$\sigma_F = \sqrt{\frac{\sum_{i=1}^{N}(F_i - \bar{F})^2}{N}}$$

$$vc = \frac{\sigma_F}{\bar{F}}$$

- Juilland's *D* and *U* – measures of dispersion (*D*) and adjusted frequency (*U*). The values are calculated using an adjusted formula, which takes differences in text sizes into account. The values of *D* fall within the range of $[0, 1]$, whereas the values of *U* fall within the range of $[0, F]$.

$$\sigma_P = \sqrt{\frac{\sum_{i=1}^{N}(P_i - \bar{P})^2}{N}}$$

$$D_{adj} = 1 - \frac{\sigma_P}{\bar{P}\sqrt{N-1}}$$

$$U = D_{adj} \cdot F$$

*Note that the standard deviation values used to calculate Juilland's D and U are calculated using different formulas than for the variation coefficient.

- Gries' *DP* (reverted *DP*, normalized *DP*) – measures of dispersion. Except for *DP*, values of reverted *DP* (DP_{rev}) (which can be easily compared to other measures) and normalized *DP* (DP_{norm}) (which can be used to compare corpora consisting of different numbers of texts) are calculated. Higher values of *DP* and normalized *DP* (and lower values of reverted *DP*) represent a higher dispersion of a given word. The values of *DP* fall within the range of $[0, 1 - min\{S_i\}_{i=1}^{N}]$, the values of normalized *DP* fall within the range of $[0, 1]$, and the values of reverted *DP* fall within the range of $[min\{S_i\}_{i=1}^{N}, 1]$.

$$DP = \frac{\sum_{i=1}^{N} | \, fracF_iF - S_i|}{2}$$

$$DP_{rev} = 1 - DP$$

$$DP_{norm} = \frac{DP}{1 - \frac{1}{N}}$$

- Caroll's *D2* and *Um* – measures of dispersion (*D2*) and adjusted frequency (*Um*). *D2* displays values from the $[0, 1]$ range; the lower the values, the more uneven the distribution of the word. The values of *Um* fall within the range of $[\frac{F}{N}, F]$.

$$D_2 = \frac{-\sum_{i=1}^{N}\left(\frac{P_i}{\sum_{i=1}^{N}P_i} \cdot \log_2 \frac{P_i}{\sum_{i=1}^{N}P_i}\right)}{\log_2 N}$$

$$U_m = F \cdot D_2 + (1 - D_2) \cdot \frac{F}{N}$$

- KL Divergence (*KLD*) – non-symmetric measure of the difference in probability distributions, used as a dispersion measure. The measure has non-negative values, where higher values represent a more uneven distribution of the word. In the following formula it is assumed that
$log_2 0 = 0$.

$$KLD = \sum_{i=1}^{N} \left(\frac{F_i}{F} \cdot \log_2 \left(\frac{F_i}{F} \cdot \frac{1}{S_i} \right) \right)$$

- Rosengren's *S* and *AF* – measures of dispersion (*S*) and adjusted frequency (*AF*). The values of the metrics are calculated using an adjusted formula, which takes differing text sizes into account. *S* displays values from the range $[\frac{1}{N}, 1]$, where higher values represent a more even distribution of the word in the corpus. The values of AF fall within the range of $[\frac{F}{/}N, F]$.

$$S_{adj} = \frac{1}{F} \left(\sum_{i=1}^{N} \sqrt{F_i \cdot S_i} \right)^2$$

$$AF = F \cdot S_{adj}$$

- *ARF* – a measure of adjusted frequency based on the distances between the occurrences of a given word. In the formula below, the variable d_j stands for the distance between the j-th and j+1-th occurrence of the word (for $j = F$ – the distance between the first and the last word, assuming that the distance between the first and the last word of the corpus is 1). The values of ARF fall within the range of $[1, F]$ (higher values represent a more even distribution of the word).

$$ARF = \frac{F}{L} \sum_{i=1}^{F} min \left\{ d_i, \frac{L}{F} \right\}$$

For corpora consisting of only one text, only frequency and ARF are calculated.

The measures of dispersion are described in more detail in „Dispersions and adjusted frequencies in corpora" (Gries, 2008) and in chapter 5 of the handbook „A Practical Handbook of Corpus Linguistics" (Gries, 2021).

1.4.2 Terminology

Terminology is generated through the TermoPL application, where you can also find its detailed description, instructions, and additional information.

The information generated by the Terminology functionality is limited to base forms, C-values, and numbers of occurrences, sorted by C-value. After clicking on the "Download" button, a txt file containing all the data generated by TermoPL is downloaded to the user's computer.

The files generated by Korpusomat are compatible with the TermoPL application - after downloading the "corpus source files" (see the main corpus panel), you can run the TermoPL application with the options of your choice.

1.5 Tools

Korpusomat employs two high-level programming libraries for natural language processing: spaCy and Stanza as well as language-specific models built by their creators.

It also employs the following tools and resources:

- Universal Dependencies,

- Marciniak, M., Mykowiecka, A., & Rychlik, P. (2016). TermoPL - a Flexible Tool for Terminology Extraction. LREC.
- Matthijs Brouwer, Hennie Brugman and Marc Kemps-Snijders 2017. MTAS: A Solr/Lucene based multi tier annotation search solution. Selected papers from the CLARIN Annual Conference 2016. Linköping Electronic Conference Proceedings 136: 19–37.

1.6 Licence

1.7 Referencing Korpusomat

W przypadku użycia w pracy naukowej prosimy o zacytowanie artykułu:

Karol Saputa, Aleksandra Tomaszewska, Natalia Zawadzka-Paluektau, Witold Kieraś, and Łukasz Kobyliński. **Korpusomat.eu: A multilingual platform for building and analysing linguistic corpora.** In Jiří Mikyška, Clélia de Mulatier, Maciej Paszynski, Valeria V. Krzhizhanovskaya, Jack J. Dongarra, and Peter M.A. Sloot, editors, Computational Science – ICCS 2023. 23rd International Conference, Prague, Czech Republic, July 3–5, 2023, Proceedings, Part II, number 14074 in Lecture Notes in Computer Science, pages 230–237, Cham, 2023. Springer Nature Switzerland. bibtex doi

1.8 Authors

1.8.1 Team

Witold Kieraś
Product Owner

Łukasz Kobyliński
Project Manager, author of the original version

Karol Saputa
Main programmer

Sandra Penno
Programmer

Filip Koselski

Programmer

1.8.2 Former collaborators

- Zbigniew Gawłowicz
- Michał Wasiluk
- Agnieszka Olech
- Filip Karpiński
- Marcel Kawski
- Michał Modzelewski
- Kacper Mirowski
- Ryszard Tuora