Orange3 Textable Prototypes Documentation

Release 0.14

University of Lausanne

Jun 04, 2020

Contents

Welcome to the documentation of Orange3 Textable Prototypes.

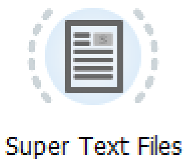
Orange3 Textable Prototypes is an open-source add-on for Orange Canvas 3. It collects text-analytic widgets that couldn't be included in the core Textable distribution for various reasons (notably because they introduce dependencies to third-party packages, or simply because they're not yet production-ready), but that may still be found useful by Orange Canvas/Textable users.

The project is brought to the community by the department of language and information sciences (SLI) at the University of Lausanne. It is hosted at https://github.com/axanthos/orange3-textable-prototypes, and the documentation can be found at http://orange3-textable-prototypes.readthedocs.org/.

CHAPTER 1

Widgets

1.1 SuperTextFiles



Import raw text, PDF and image files with if necessary an usage of Tesseract, an OCR application.

1.1.1 Authors

Loïc Aubrays, Fábio Torres Cabral (Aris Xanthos, original Text Files)

1.1.2 Signals

Inputs: `Message`

> JSON Message controlling the list of imported text files

Outputs: `Text data`

> Segmentation covering the content of imported text files

1.1.3 Installation of Tesseract

To use the OCR feature, an extra installation is needed. There are two parts to install, the engine itself, and the training data for a language. Please read the official documentation of Tesseract.

1.1.4 Description

This widget is designed to import the contents of one or several files in Orange Canvas. It outputs a segmentation containing a (potentially annotated) segment for each imported file.

This widget processes files with 3 methods.

For raw text files

The imported textual content is normalized in several ways:

- it is systematically converted to Unicode (from the encoding defined by the user)
- it is subjected to the canonical Unicode decomposition-recomposition technique (Unicode sequences such as `LATIN SMALL LETTER C (U+0063)` + `COMBINING CEDILLA (U+0327)` are systematically replaced by the combined equivalent, e.g. `LATIN SMALL LETTER C WITH CEDILLA (U+00C7)`)
- it is stripped from the utf8 byte-order mark (if any)
- various forms of line endings (in particular `\r\n` and `\r`) are converted to a single form (namely `\n`)

For textual PDF files

The textual content is extracted from the PDF file in the order in the file, not in the reading view.

For pictures and PDF files

The textual content is extracted from the images with the Tesseract OCR

The interface of **Super Text files** is available in two versions, according to whether or not the **Advanced Settings** checkbox is selected.

Basic interface

In its basic version (see *figure 1* below), the **Text Files** widget is limited to the import of a single text or textual PDF file. The interface contains a **Source** section enabling the user to select the input file. The **Browse** button opens a file selection dialog; the selected file then appears in the **File path** text field (it can also be directly inputted with the keyboard). The **Encoding** drop-down menu enables the user to specify the encoding of the file.

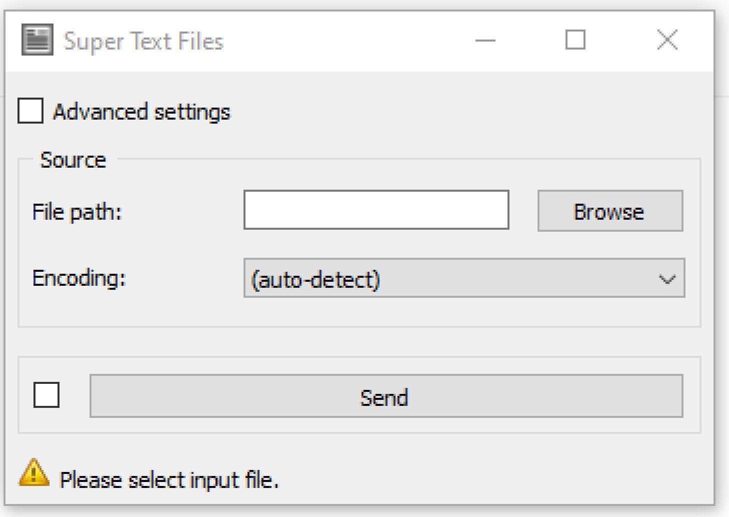
Note that the language is assumed to be English for OCR purposes (it can be specified otherwise using the advanced interface.

The **Send** button triggers the emission of a segmentation to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

The text below the **Send** button indicates the number of characters in the single segment contained in the output segmentation, or the reasons why no segmentation is emitted (no input data, encoding issue, etc.).

Advanced interface

The advanced version of **Super Text Files** allows the user to import several files in a determined order; each file can moreover be associated to a distinct encoding and specific annotations. The emitted segmentation contains a segment for each imported file.



Encoding: (auto-detect)

⚠ Please select input file.

Fig. 1: Figure 1: **Super Text files** widget (basic interface).

The advanced interface (see *figure 2* above) presents similarities with that of the **URLs**, **Recode**, and **Segment** widgets. The **Sources** section allows the user to select the input file(s) as well as their encoding, to determine the order in which they appear in the output segmentation, and optionally to assign an annotation. The list of imported files appears at the top of the window; the columns of this list indicate (a) the name of each file, (b) the corresponding annotation (if any), and (c) the encoding with which each is associated.

The first buttons on the right of the imported files' list enable the user to modify the order in which they appear in the output segmentation (**Move Up** and **Move Down**), to delete a file from the list (**Remove**) or to completely empty it (**Clear All**). Except for **Clear All**, all these buttons require the user to previously select an entry from the list. **Import List** enables the user to import a file list in JSON format (see sections *JSON im-/export format* and *File list* in Textable documentation) and to add it to the previously selected sources. In the opposite **Export List** enables the user to export the source list in a JSON file.

The remainder of the **Sources** section allows the user to add new files to the list. The easiest way to do so is to first click on the **Browse** button, which opens a file selection dialog. After having selected one or more files in this dialog and validated the choice by clicking on **Open**, the files appear in the **File paths** field and can be added to the list by clicking on the **Add** button. It is also possible to type the complete paths of the files directly in the text field, separating the paths corresponding to the successive files with the string ” ” (space + slash + space).

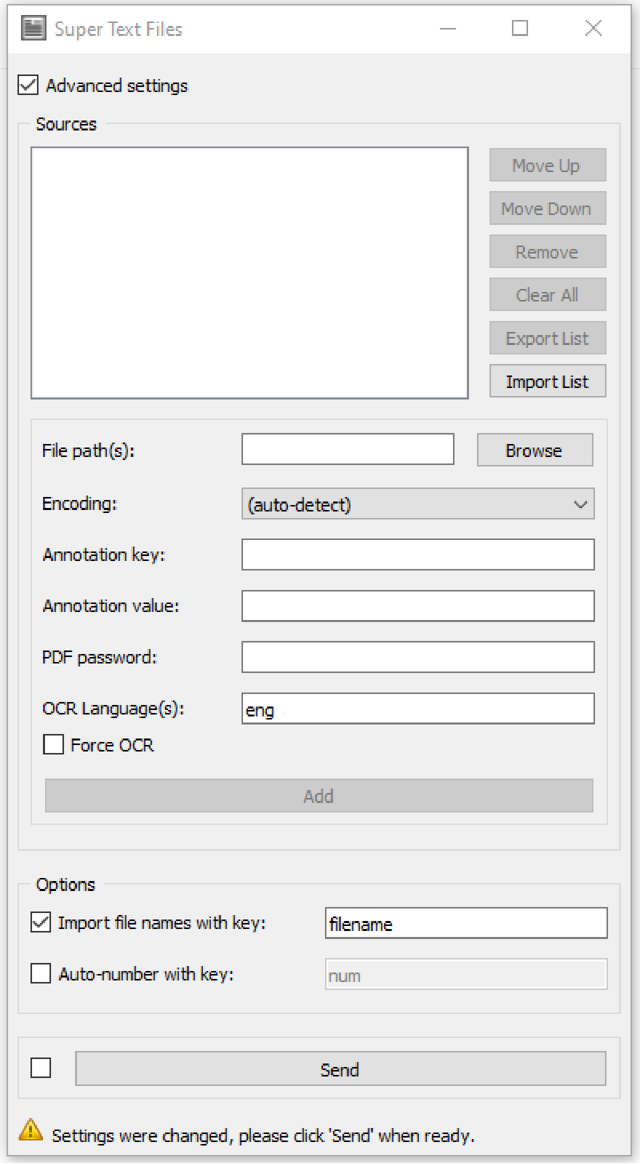
Before adding one or more files to the list by clicking on **Add**, it is possible to select their encoding (**Encoding**), and to assign an annotation by specifying its key in the **Annotation key** field and the corresponding value in the **Annotation value** field. These three parameters (encoding, key, value) will be applied to each file appearing in the **File paths** field at the moment of their addition to the list with **Add**.

The **PDF Password** field allows password-protected files to be passed to the widget. Insert the password in the field and proceed as usual.

The **OCR Language(s)** field is needed by the OCR processes. By default, it contains English abbreviated by *eng*. Multilingual files are supported by complete the field with the languages separated by +. For example, *eng+fra+ita*. See all installable Tesseract data files. If nothing is indicated in this field, the language is assumed to be *eng*.

Force OCR enables the OCR processes of the widget while also extracting textual content. Enable this if textual and image content are both present in the file (or if extraction of textual content gives really bad results).

The **Options** section allows the user to specify the label affected to the output segmentation. The **Import filenames with key** checkbox enables the program to create for each imported file an annotation whose value is the file name (as displayed in the list) and whose key is specified by the user in the text field on the right of the checkbox. Similarly the



☑ Advanced settings

File path(s):

Encoding: (auto-detect)

Annotation key:

Annotation value:

PDF password:

OCR Language(s): eng

☐ Force OCR

☑ Import file names with key: filename

☐ Auto-number with key: num

☐

Settings were changed, please click 'Send' when ready.

Fig. 2: Figure 2: **Super Text files** widget (advanced interface).

button **Auto-number with key** enables the program to automatically number the imported files and to associate the number to the annotation key specified in the text field on the right.

The **Send** button triggers the emission of a segmentation to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

The text below the **Send** button indicates the length of the output segmentation in characters, or the reasons why no segmentation is emitted (no selected file, encoding issue, etc.). In the example, the two segments corresponding to the imported files thus total up to 1‘262‘145 characters.

Remote control

Super Text Files is one the widgets that can be controlled by means of the **Message** widget. Indeed, it can receive in input a message consisting of a file list in JSON format (see sections *JSON im-/export format* and *File list* in Textable documentation), in which case the list of files specified in this message replaces previously imported sources (if any). Note that removing the incoming connection from the **Message** instance will not, by itself, remove the list of files imported in this way from the **Super Text Files** instance's interface; conversely, this list of files can be modified using buttons **Move up/down**, **Remove**, etc. even if the incoming connection from the **Message** instance has not been removed. Finally, note that if a **Super Text Files** instance has the basic version of its interface activated when an incoming connection is created from an instance of **Message**, it automatically switches to the advanced interface.

1.1.5 Messages

Information

Data correctly sent to output: <n> segments (<m> characters). This confirms that the widget has operated properly.

Settings were* (or *Input has*) *changed, please click ‘Send’ when ready. Settings and/or input have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

No data sent to output yet: no file selected. The widget instance is not able to emit data to output because no input file has been selected.

No data sent to output yet, see ‘Widget state’ below. A problem with the instance's parameters and/or input data prevents it from operating properly, and additional diagnostic information can be found in the **Widget state** box at the bottom of the instance's interface (see *Warnings* and *Errors* below).

Warnings

No label was provided. A label must be entered in the **Output segmentation label** field in order for computation and data emission to proceed.

No annotation key was provided for auto-numbering. The **Auto-number with key** checkbox has been selected and an annotation key must be specified in the text field on the right in order for computation and data emission to proceed.

JSON message on input connection doesn't have the right keys and/or values. The widget instance has received a JSON message on its `Message` input channel and the keys and/or values specified in this message do not match those that are expected for this particular widget type (see sections *JSON im-/export format* and *File list* in Textable documentation).

JSON parsing error. The widget instance has received data on its `Message` input channel and the data couldn't be correctly parsed. Please use a JSON validator to check the data's well-formedness.

Errors

Couldn't open file* or *Couldn't open file '<filepath>'. A file couldn't be opened and read, typically because the specified path is wrong.

Encoding error* or *Encoding error: file '<filepath>'. A file couldn't be read with the specified encoding (it must be in another encoding).

Please make sure all Tesseract parameter files for language(s) '<languages>' have been installed.. One or more Tesseract language packages are probably missing.

Tesseract is not installed or it's not in your path. Add the directory where the tesseract-OCR binaries are located to the Environment Path variables, probably `C:\Program Files\Tesseract-OCR`

1.2 Lyrics Genius

Make a corpus with songs lyrics.

1.2.1 Author

Cyrille Gay-Crosier, Rafael Bruni Baschino, Basile Maillard

1.2.2 Signals

Input: None

Outputs:

- `Text data`

  A segmentation with the lyrics of the selected songs.

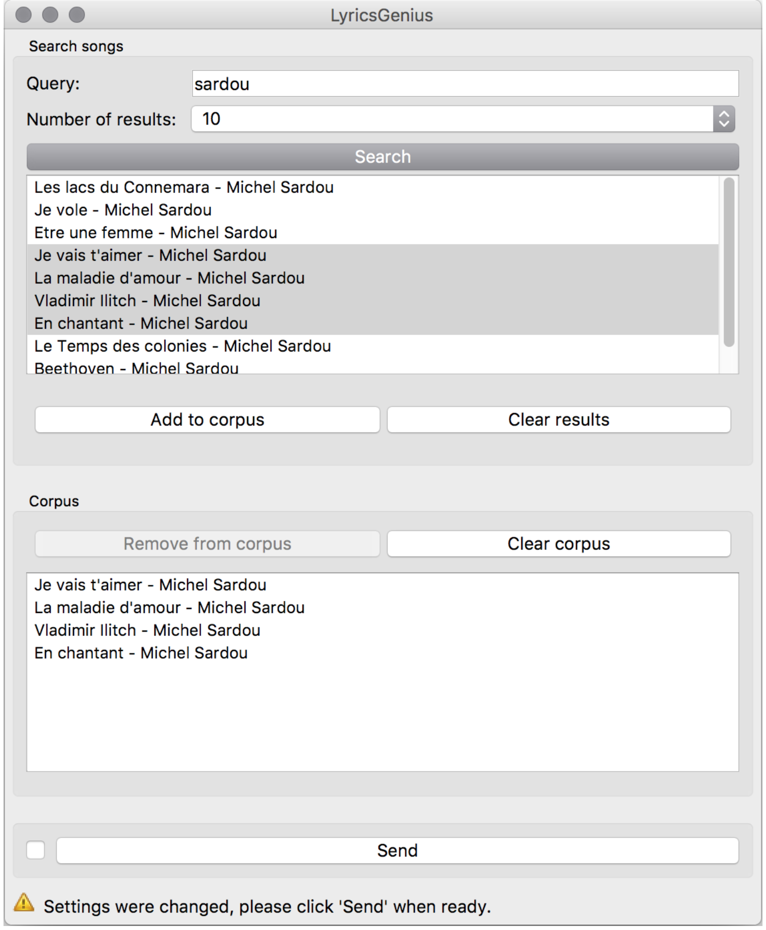
1.2.3 Description

This widget is designed to import one or more songs lyrics in Orange Canvas. The lyrics are retrieved from https://genius.com/. The output is a segmentation containing a segment for each imported songs. Each segment has 4 annotations with keys *artist*, *artist_id*, *path*, *title*.

Interface

The **Lyrics Genius** widget simply lets the user make a search on the Genius website.

The **Create my corpus** section allows the user to add or remove songs from the search results.



Query: sardou

Number of results: 10

Les lacs du Connemara - Michel Sardou
Je vole - Michel Sardou
Etre une femme - Michel Sardou
Je vais t'aimer - Michel Sardou
La maladie d'amour - Michel Sardou
Vladimir Ilitch - Michel Sardou
En chantant - Michel Sardou
Le Temps des colonies - Michel Sardou
Beethoven - Michel Sardou

Je vais t'aimer - Michel Sardou
La maladie d'amour - Michel Sardou
Vladimir Ilitch - Michel Sardou
En chantant - Michel Sardou

□ Send

⚠ Settings were changed, please click 'Send' when ready.

Fig. 3: Figure 1: **Lyrics Genius** widget interface.

The **My titles** section is the user "basket". He can add or remove songs from his corpus with the section **Create my corpus**.

The **Send** button triggers the emission of a segmentation to the output connection(s). When selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

1.2.4 Messages

Information

<n> segments sent to output (<m> characters). This confirms that the widget has operated properly.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

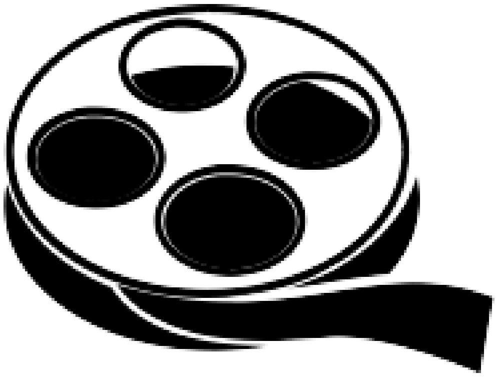
You didn't search anything The user want to make a search with no text in the query box.

Your corpus is empty, please add some songs first The corpus is empty, so the user have to add some songs before sending.

Errors

Couldn't download data from Genius website. An error has prevented the widget to download the data from the Genius website (most likely related to a connection problem).

1.3 Movie Transcripts



Make a corpus of movie transcripts

1.3.1 Authors

Leonardo Cavaliere, David Flühmann, Kirill Melnikov

1.3.2 Signals

Input: None

Outputs:

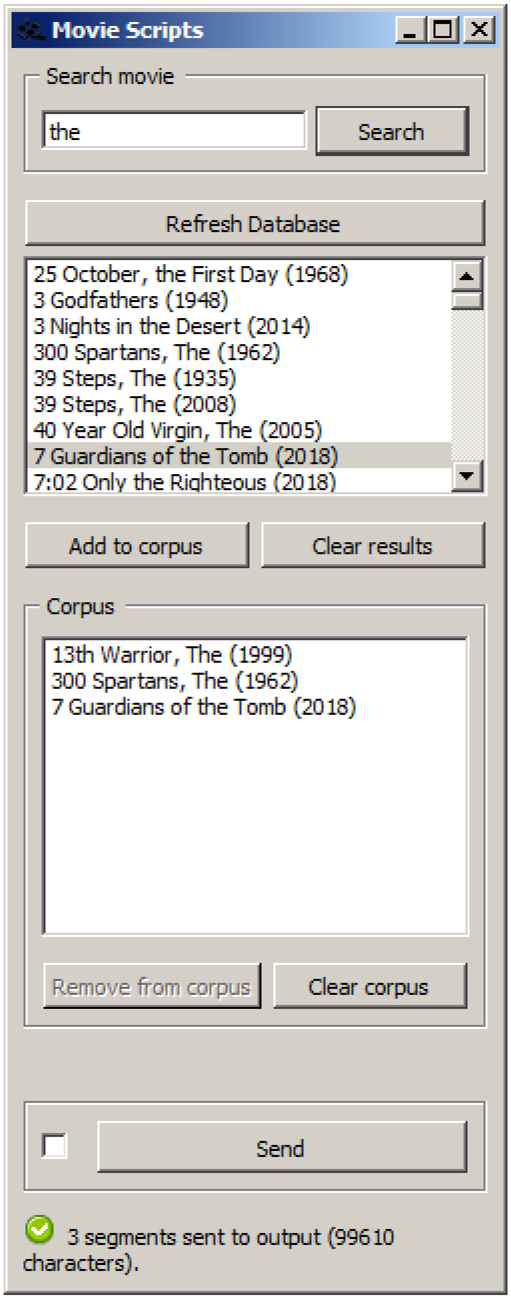
- `Text data`

  A segmentation with the transcripts of the selected movies.

1.3.3 Description

This widget is designed to import one or more movie transcripts in Orange Canvas. The transcripts are retrieved from https://www.springfieldspringfield.co.uk/. The output is a segmentation containing a segment for each imported movie.

Interface



Search movie: the

25 October, the First Day (1968)
3 Godfathers (1948)
3 Nights in the Desert (2014)
300 Spartans, The (1962)
39 Steps, The (1935)
39 Steps, The (2008)
40 Year Old Virgin, The (2005)
7 Guardians of the Tomb (2018)
7:02 Only the Righteous (2018)

Corpus:

13th Warrior, The (1999)
300 Spartans, The (1962)
7 Guardians of the Tomb (2018)

□ Send

3 segments sent to output (99610 characters).

Figure 1: **Movie transcripts** widget interface

The **Movie transcripts** widget simply lets the user make a search on the SpringfieldSpringfield website.

The **Search** button searches the database for entries matching user's input.

The **Refresh database** button downloads the newest collection of all the movie titles available on the website and stores them in cache.

The **Corpus** section is a container where the user's transcripts are stored. The user can add or remove transcripts to and from their corpus or clear the corpus entirely.

The **Send** button triggers the emission of a segmentation to the output connection(s). When selected, the Send automatically checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

1.3.4 Messages

Information

<n> segments sent to output (<m> characters). This confirms that the widget has operated properly.

Database successfully updated This confirms that the Refresh button worked properly.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Please, enter a query in the search bar The user attempted to make a search with no text in the query box.

Your corpus is empty, please add some movies first The corpus is empty, no movies have been selected for downloading.

Errors

Couldn't download data from SpringfieldSpringfield website. An error has prevented the widget from downloading the data from the SpringfieldSpringfield website (most likely related to a connection problem).

Couldn't save database on disk An error has prevented the widget from saving the downloaded database on the tser's hard drive.

1.4 Theatre Classique

Import theater plays from the theatre-classique website (in TEI-XML).

1.4.1 Author

Aris Xanthos

1.4.2 Signals

Inputs: None

Outputs:

- `Text data`

  Segmentation covering the content of imported TEI-XML-encoded theatre plays

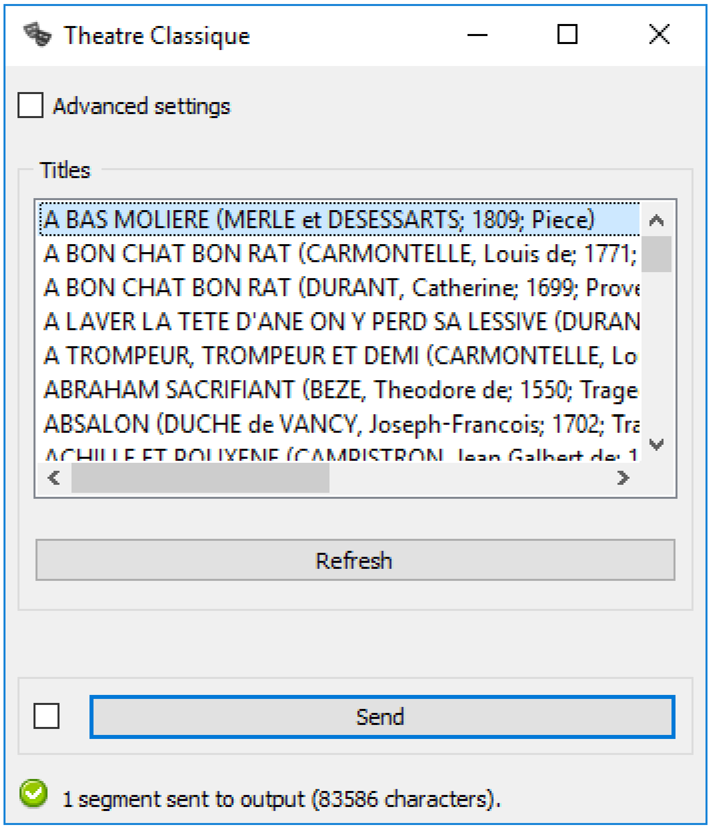
1.4.3 Description

This widget is designed to import one or more theatre plays in Orange Canvas. The plays are retrieved from http://www.theatre-classique.fr and richly encoded in TEI-XML format. The output is a segmentation containing a segment for each imported play. Each segment has 5 annotations with keys *author*, *title*, *year*, *genre*, and *url*.

The interface of **Theatre Classique** is available in two versions, according to whether or not the **Advanced Settings** checkbox is selected.

Basic interface

In its basic version (see *figure 1* below), the **Theatre Classique** widget simply lets the user select one or more plays in the catalogue of more than 800 entries downloadable from the theatre-classique website. To select multiple files use either control/command-click or shift-click.



A BAS MOLIERE (MERLE et DESESSARTS; 1809; Piece)
A BON CHAT BON RAT (CARMONTELLE, Louis de; 1771;
A BON CHAT BON RAT (DURANT, Catherine; 1699; Prove
A LAVER LA TETE D'ANE ON Y PERD SA LESSIVE (DURAN
A TROMPEUR, TROMPEUR ET DEMI (CARMONTELLE, Lo
ABRAHAM SACRIFIANT (BEZE, Theodore de; 1550; Trage
ABSALON (DUCHE de VANCY, Joseph-Francois; 1702; Tra
ACHILLE ET POLIXENE (CAMPISTRON, Jean Galbert de; 1

1 segment sent to output (83586 characters).

Fig. 4: Figure 1: **Theatre Classique** widget (basic interface).

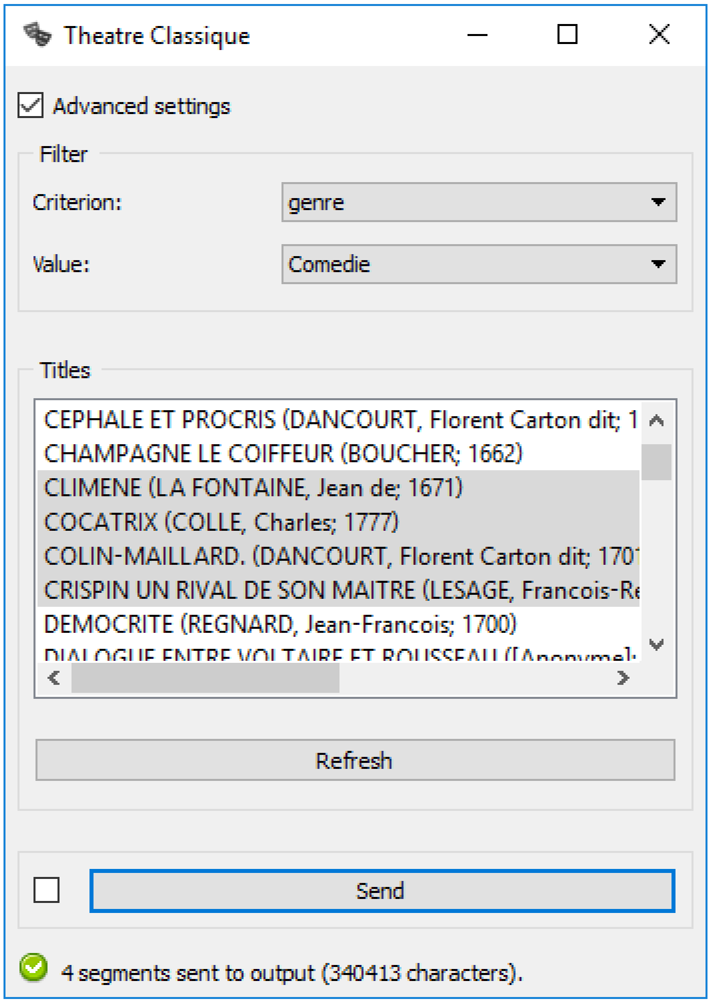
The **Options** section allows the user to define the label of the output segmentation (**Output segmentation label**).

The **Info** section indicates the number of segments and characters in the output segmentation, or the reasons why no segmentation is emitted (no title selected, connection issues, etc.).

The **Send** button triggers the emission of a segmentation to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

Advanced interface

The advanced version of **Theatre Classique** (see *figure 2* below)offers the same functionality as the basic one, and it adds the possibility of selecting only the plays of a given author/genre/title.



☑ Advanced settings

Criterion: genre

Value: Comedie

CEPHALE ET PROCRIS (DANCOURT, Florent Carton dit; 1
CHAMPAGNE LE COIFFEUR (BOUCHER; 1662)
CLIMENE (LA FONTAINE, Jean de; 1671)
COCATRIX (COLLE, Charles; 1777)
COLIN-MAILLARD. (DANCOURT, Florent Carton dit; 170
CRISPIN UN RIVAL DE SON MAITRE (LESAGE, Francois-Re
DEMOCRITE (REGNARD, Jean-Francois; 1700)
DIALOGUE ENTRE VOLTAIRE ET ROUSSEAU ([Anonyme];

□ Send

4 segments sent to output (340413 characters).

Fig. 5: Figure 2: **Theatre Classique** widget (advanced interface).

The **Options** and **Info** sections, as well as the **Send** button and **Send automatically**, operate in the same way as in the basic interface.

1.4.4 Messages

Information

<n> segments sent to output (<m> characters). This confirms that the widget has operated properly.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Please select one or more titles. The widget instance is not able to emit data to output because no theatre play has been selected.

Errors

Couldn't download data from theatre-classique website. An error has prevented the widget to download the data from the theatre-classique (most likely related to a connection problem).

1.5 Extract CSV

Extract tabulated data as a Textable Segmentation

1.5.1 Author

Sorcha Walsh, Noémie Carette, Saara Jones

1.5.2 Signals

Inputs: Tabulated data

Outputs:

- `Text data`

  Segmentation of tabulated data

1.5.3 Description

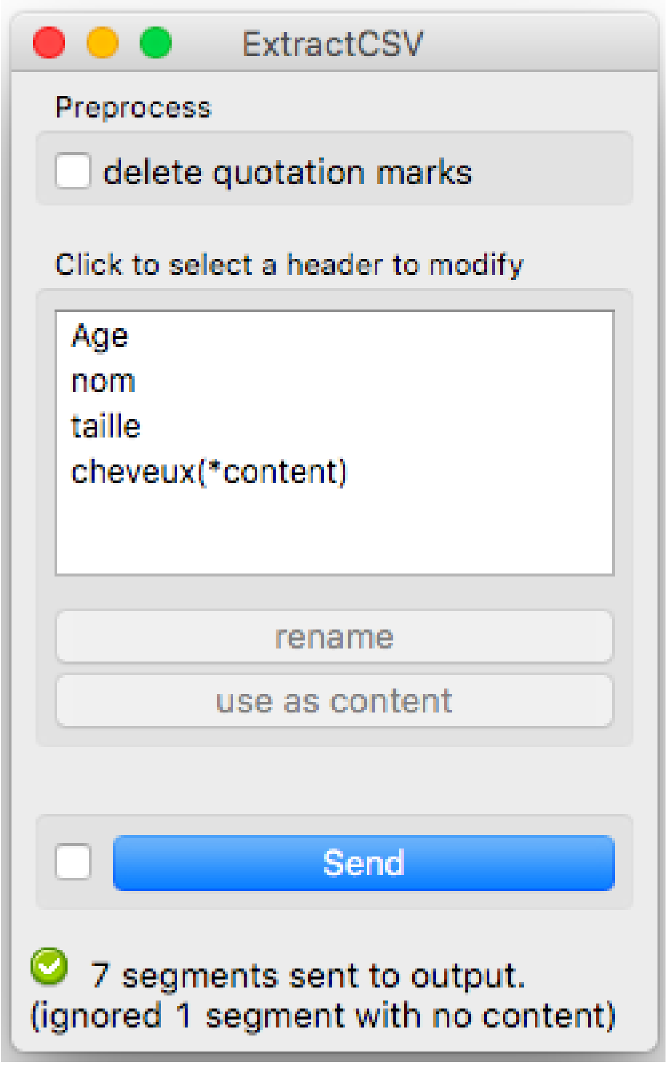
This widget is designed to extract tabulated data as a Textable Segmentation in Orange Canvas. It takes .csv files as input and the user can choose which header he wants to use as content. The output is a segmentation containing a segment for each line of the file and each segment has annotations with each column's header as keys.

Interface

Extract CSV displays a list of the file's headers and lets the user choose which one to use as content for the segmentation with the **Use as Content** button. If the file has no header it will simply display the number of the column.

Quotation marks usually cause problems with detecting the segment position. If the file has quotes, the user can choose to preprocess his file with the checkbox **delete quotation marks**.

The **Rename** button allows the user to rename the headers.



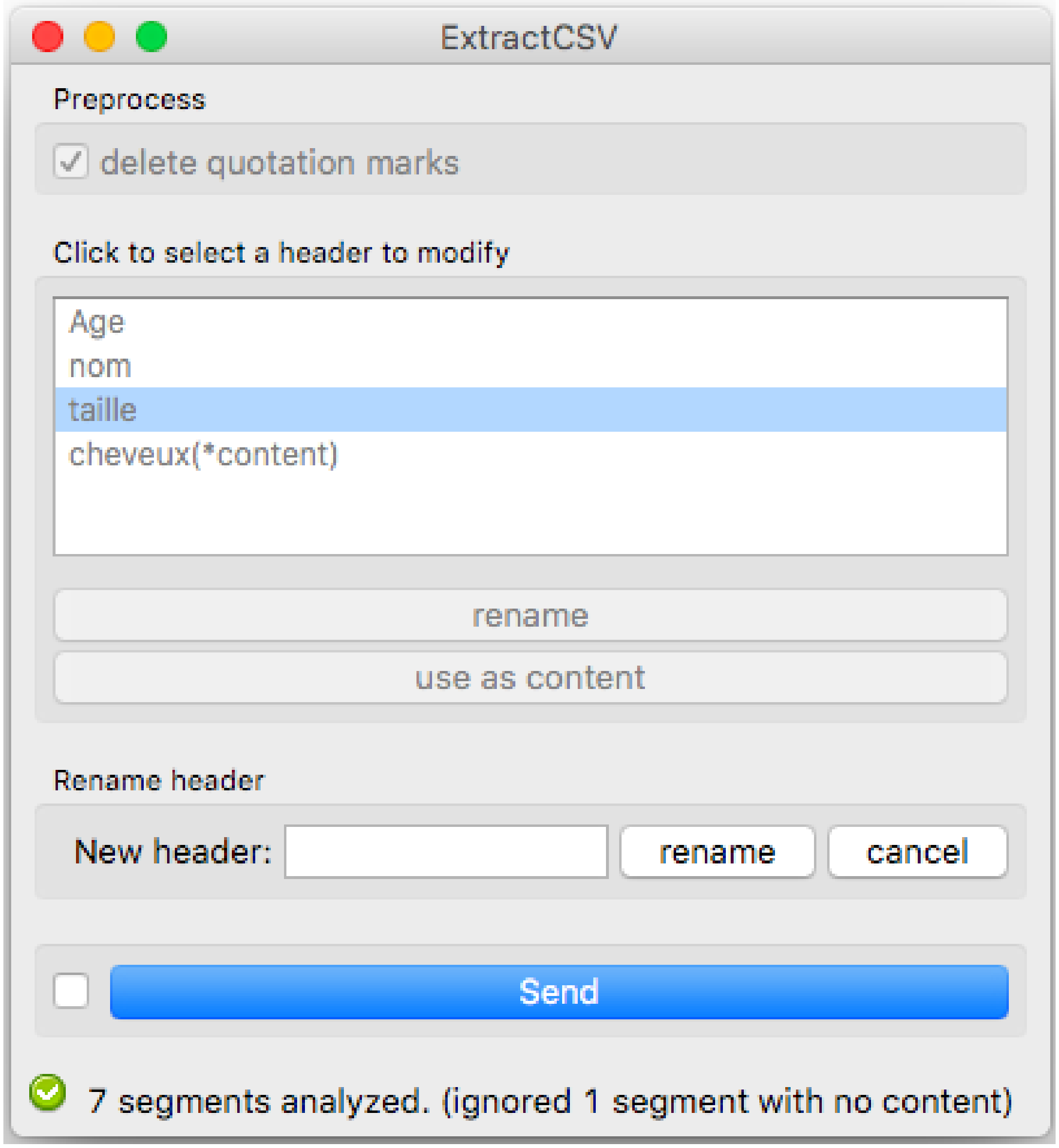
□ delete quotation marks

Age
nom
taille
cheveux(*content)

□ Send

✓ 7 segments sent to output.
(ignored 1 segment with no content)

Fig. 6: Figure 1: **Extract CSV** widget



Preprocess

☑ delete quotation marks

Click to select a header to modify

Age
nom
taille
cheveux(*content)

Rename header

New header:

□ Send

7 segments analyzed. (ignored 1 segment with no content)

Fig. 7: Figure 2: **Extract CSV** widget

The **Info** section indicates the number of segments in the output segmentation, or if any segment has been ignored because of a missing content.

The **Send** button triggers the emission of a segmentation to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

1.5.4 Messages

Information

<n> segments analyzed. This confirms that the widget has operated properly.

<n> segments sent to output. (Ignored <m> segments with no content) Informs the user if segments were ignored.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

1.6 Redditor

1.6.1 Authors

Loris Rimaz, Olivia Edelman, Nahuel Degonda

1.6.2 Signals

Inputs: None

Outputs:

- `Segmentation`

  Segmentation with segments for each post and comment in imported corpora

1.6.3 Description

This widget is designed to scrap one or more different types of content from Reddit, data is retrieved from <*https://www.reddit.com/*>. The widget outputs at least one segmentation with the the title and content of a post or subreddit, with the possibility to include or exclude comments and images. Segments in this segmentation have a number of annotations :

KEY	EXAMPLE VALUE
Author	*cosmicnebula257*
Id	*bp1ze4*
Parent	*bp1ze4*
Parent_type	*0*
Posted_Unix	*1557946951.0*
Posted_at	*2009-5-15 19:02:31*
Score	*1*
Title	*Random invites while AFK in the menu*

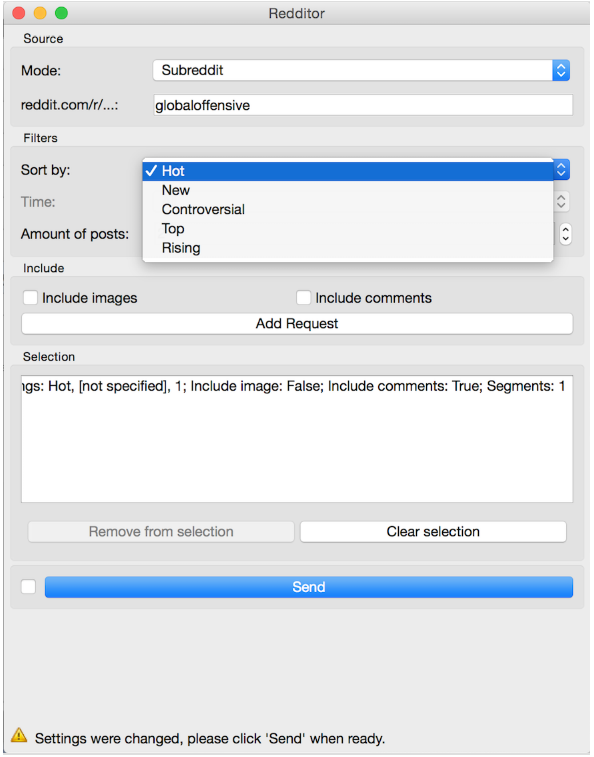
1.6.4 Interface

The **Redditor** widget lets the user select one or more posts or subreddits from the Reddit website‘<https://www.reddit.com/>‘.

The **Subreddit** mode allows the user to scrap the data of an entire subreddit. Users can sort the subreddits by:

- *Hot*
- *New*
- *Controversial*
- *Top*
- *Rising*

Users can also choose the amount of posts they want to upload to the widget.



Mode: Subreddit

reddit.com/r/...: globaloffensive

Sort by: ✓ Hot

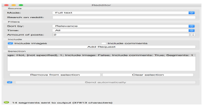
□ Include images □ Include comments

ıgs: Hot, [not specified], 1; Include image: False; Include comments: True; Segments: 1

⚠ Settings were changed, please click 'Send' when ready.

The **Full Text** mode allows the user to upload/scrap from all of Reddit and not just specifically from a subreddit or a url. The **Full Text** mode has three filters.

1. The **Sort by** filter : *Relevance*, *Top*, *New*, *Comments*.
2. The **Time** filter : *All*, *Past day*, *Past hour*, *Past month*, *Past hour*.
3. The **Amount of posts** filter, just like in the **Subreddit** mode.



Mode: Full text

Search on reddit:

Sort by: Relevance

Time: All

Amount of posts: 2

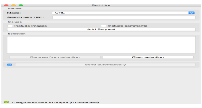
☑ Include images □ Include comments

…gs: Hot, [not specified], 1; Include image: False; Include comments: True; Segments: 1

☑ Send automatically

14 segments sent to output (37813 characters)

The **URL** mode allows the user to search data by directly using the URL of a post or subreddit.



Mode: URL

Search with URL:

□ Include images □ Include comments

☑ Send automatically

0 segments sent to output (0 characters)

The widget also has options available for all three modes.

The **Include Images** option allows the user to choose wether or not he wants to include the images of certain posts and/or subreddits.

The **Include Comments** option allows the user to choose wether or not he wants to include comments in his output.

□ Include images □ Include comments

The **Add Request** button allows users to add multiple posts and/or subreddits to the basket in the **Selection** box. To select multiple files use the **Selection** box to add them to the basket.

The **Send** button is used to send data to output.

When the widget gets closed and re-opened, the content of it is saved. To delete this saved content, the user has to reset the widget settings.

1.6.5 Messages

Information

<n> segments sent to output. This confirms that the widget has operated properly.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

The post found only contains images. Try to include images or comments. The widget instance is unable to emit image to output because the user hasn't checked the box for images

Please fill in the input box. The widget instance is unable to emit data to output because the user hasn't filled in the input box.

Errors

Error in redirect, please make sure the subreddit name is correct. An error has prevented the widget to download the data from Reddit, most likely because of a misspelling.

Subreddit not found. An error has prevented the widget to download the data from Reddit, the subreddit was not found, might not exist, might be a user mistake.

No match for URL. An error has prevented the widget to download the data from Reddit, the URL was not found, might not exist, might be a user mistake.

URL not found. An error has prevented the widget to download the data from Reddit, the input was not in URL format.

1.7 CHILDES

Import data in XML format from the CHILDES database.

1.7.1 Author

Aris Xanthos

1.7.2 Signals

Inputs: None

Outputs:

- `Files`

  Segmentation with a segment for each file in imported corpora

- `Utterances` (optional)

  Segmentation with a segment for each utterance in imported corpora

- `Words` (optional)

  Segmentation with a segment for each word in imported corpora

1.7.3 Description

This widget is designed to import one or more CHILDES corpora in Orange Canvas. The corpora are retrieved from https://childes.talkbank.org/data-xml/ and richly encoded in XML format. The widget outputs at least one segmentation containing a segment for each file in each imported corpus. Segments in this segmentation have a variable number of annotations (depending on what is available for each corpus):[1]

[1] The user is referred to the CHAT transcription format documentation for the meaning and possible values of the numerous annotations extracted by the widget.

key	example value
corpus	*Geneva*
file_path	*Geneva/020107.xml*
lang	*fra*
pid	*11312/c-00028161-1*
target_child_id	*CHI*
target_child_age	*P2Y01M07D*
target_child_days	*757*
target_child_months	*25*
target_child_years	*2*

Optionally, the output may also include two more segmentations, into utterances and into words. Both inherit the annotations above. The utterance segmentation adds two extra annotations:

key	example value
uID	*u0*
who	*CHI*

The word segmentation inherits all previous annotations, and adds a variable number of annotations (depending on the information available in the data), namely at most:

key	example value
head	*1*
index	*0*
pos	*part*
prefixes	*dé*
relation	*OBJ*
stem	*faire*
suffixes	*PP&m*

Interface

User controls are divided into three main sections (see *figure 1* below): **Browse database**, **Selection**, and **Options**.

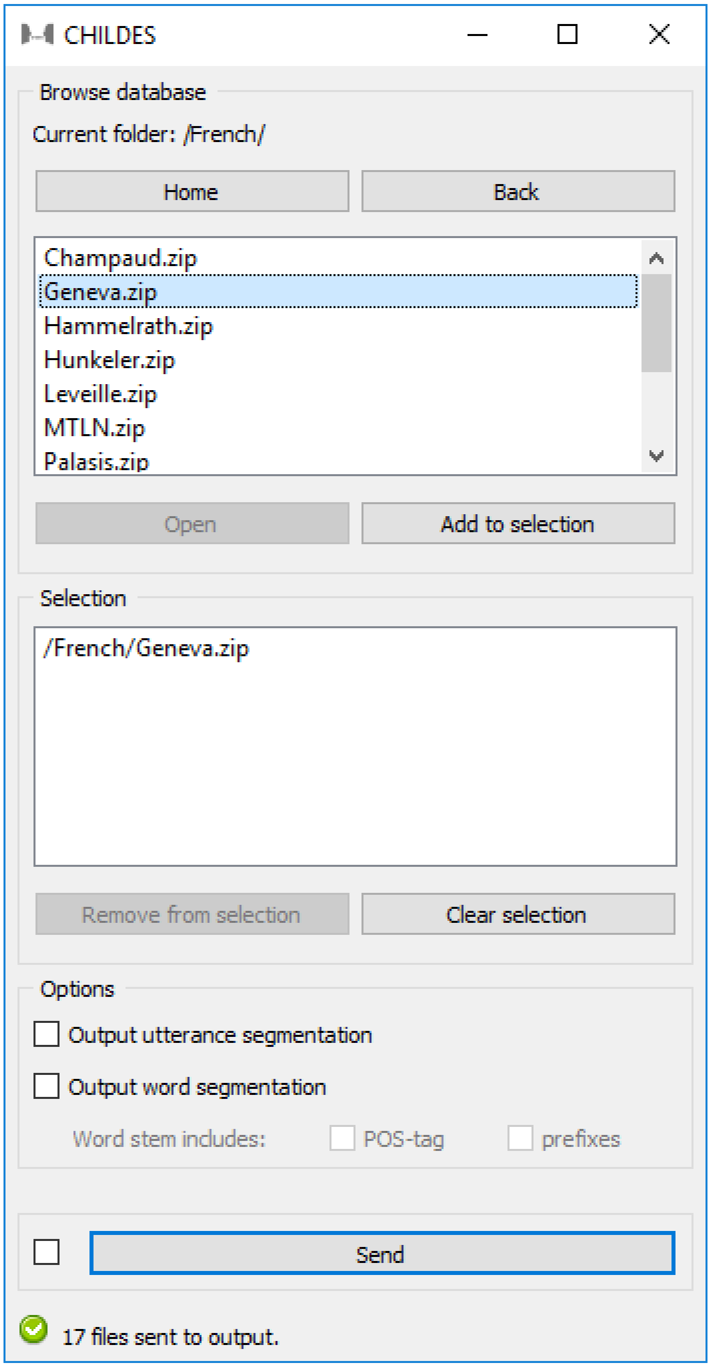
The **Browse database** section allows the user to navigate the XML section of the CHILDES database (https://childes.talkbank.org/data-xml/) and select the desired corpora. It is organized like a file tree, starting from a root folder (denoted as "/"), and each folder may contain any number of subfolders and/or zipped archives.

To view the contents of a folder, either double-click it or select it and click **Open**. Button **Back** brings you back to the parent folder, and **Home** to the root folder.

Add to selection adds the highlighted archive(s) to your selection, and so does double-clicking an archive. If a folder is highlighted, clicking **Add to selection** results in adding *all* the archives contained in this folder and, recursively, in contained subfolders (so possibly a lot of archives), to your selection. Note that multiple archives/folders may be highlighted (using control/command-click or shift-click) and added at once to your selection.

When the current folder is the root folder ("/"), the **Home** button is replaced with **Refresh**. Clicking **Refresh** instructs the widget to connect to the CHILDES website and update its own configuration to take into account possible changes (usually additions) to the database. This operation may take a few minutes and is only useful when the online database has changed; it has the additional consequence that it cancels previous selections.

The **Selection** section lists all corpora that are marked for import. Note that only a single copy of any given corpora can be added to this list (attempting to add it twice will have no effect).



Current folder: /French/

Champaud.zip
Geneva.zip
Hammelrath.zip
Hunkeler.zip
Leveille.zip
MTLN.zip
Palasis.zip

/French/Geneva.zip

□ Output utterance segmentation

□ Output word segmentation

Word stem includes: □ POS-tag □ prefixes

□ Send

17 files sent to output.

Fig. 8: Figure 1: **CHILDES** widget interface.

Remove from selection removes the highlighted archive(s) from your selection, and so does double-clicking an archive. Multiple archives may be highlighted (using control/command-click or shift-click) and removed at once from your selection. **Clear selection** removes *all* archives from your selection.

The **Options** section enables the user to select whether optional segmentations (utterances and words) should be extracted and sent in output. It also offers two options for fine-tuning the word extraction process: ticking **Words stem includes: POS-tag** prepends a word's *stem* annotation with its part-of-speech tag, which can be useful to separate homophonous stems such as *v|walk* and *n|walk*; **Words stem includes: prefixes** prepends stems with prefixes, if any, which is necessary if you want to treat e.g. *write* and *re#write* as separate stems.

The **Info** section informs the user about the status of the widget and indicates the number of segments and characters in the output segmentation, or the reasons why no segmentation is emitted (no corpus selected, connection issues, etc.).

The **Send** button triggers the retrieval and emission of one or more segmentations to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

Caveat about word extraction

Extracting a word segmentation based on CHILDES XML has proved a challenging task and should be considered a beta feature at this point in the development of the widget. The extraction strategy that has been implemented is based on the author's perception of the most important features of the CHILDES XML schema and has only be tested thus far on a limited fraction of the CHILDES corpora. This strategy can be summarized as follows:

1. substitute replacements (if any) for words, e.g. *lemme [: let me]*
2. move <*gra*> elements inside adjacent non-compound words (<*mc*>)
3. extract all words (<*w*>)
4. for each non-compound word (<*mw*>) in each (possibly compound) word:
    1. create a new word segment
    2. extract this non-compound word's attributes and assign them as annotations to the new word segment

Word attribute extraction operates as follows:

- if available, syntactic category and subcategories are joined with colons (*:*) to form the value of annotation *pos*
- if available, prefixes (<*mpfx*>) are joined with sharp (*#*) to form the value of annotation *prefixes*
- if available, suffixes (<*mk*>) are joined with *&*, - or *:* to form the value of annotation *suffixes*
- if available, *stem* attribute forms the value of annotation *stem* (possibly including *pos* and *prefixes*, depending on selected options)
- if available, *index*, *head* and *relation* attributes of <*gra*> elements are extracted to form corresponding annotations.

1.7.4 Messages

Information

<n> files, <m> utterances and <l> words sent to output. This confirms that the widget has operated properly.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Please add a corpus to the selection. The widget instance is not able to emit data to output because no corpus has been added to the selection.

Connecting to CHILDES website, please wait… The widget instance is in the process of connecting with the CHILDES website in order to recreate the database cache.

Errors

Couldn't download corpus %s from CHILDES website. An error has prevented the widget from downloading the indicated corpus from the CHILDES website.

Error while attempting to scrape the CHILDES website. An error has prevented the widget to scrape the data from the CHILDES website while recreating the database cache.

Couldn't save database to disk. An error has prevented the widget from saving the database cache to disk after recreating the database cache.

1.8 Linguistica

Unsupervised morphological analysis.

1.8.1 Author

Aris Xanthos

1.8.2 Signals

Input:

- `Word segmentation`

  A Textable segmentation containing words

Outputs:

- `Morphologically analyzed data`

  A Textable segmentation containing the same words annotated with the discovered morphological structure (stem, suffix, and signature)

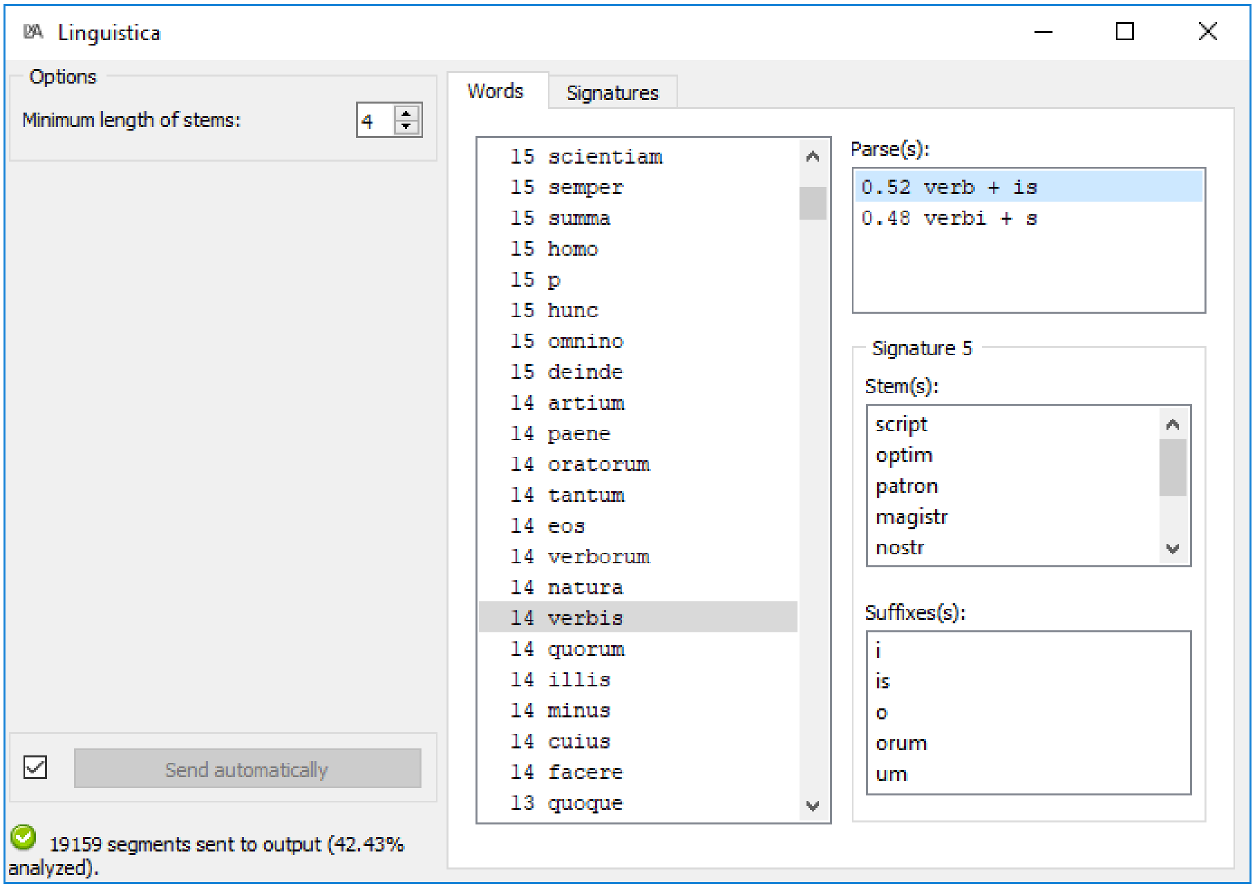
1.8.3 Description

This widget takes a word segmentation as input and applies part of John Goldsmith's "Crab Nebula" algorithm to it. The algorithm seeks to discover morphological structure in an unsupervised fashion, i.e. without using language-dependent linguistic resources.

In particular, the widget tries to divide each word into stem and a suffix, in a way that the resulting components can be regularly combined with other stems and suffixes (thus forming a structure known as a "signature").

Interface

The widget's interface displays a single control (see *figure 1* below): the user can set the minimum length allowed for a stem (4 characters is the default value).



Minimum length of stems: 4

```
15 scientiam
15 semper
15 summa
15 homo
15 p
15 hunc
15 omnino
15 deinde
14 artium
14 paene
14 oratorum
14 tantum
14 eos
14 verborum
14 natura
14 verbis
14 quorum
14 illis
14 minus
14 cuius
14 facere
13 quoque
```

Parse(s):

```
0.52 verb + is
0.48 verbi + s
```

Signature 5

Stem(s):

script
optim
patron
magistr
nostr

Suffixes(s):

i
is
o
orum
um

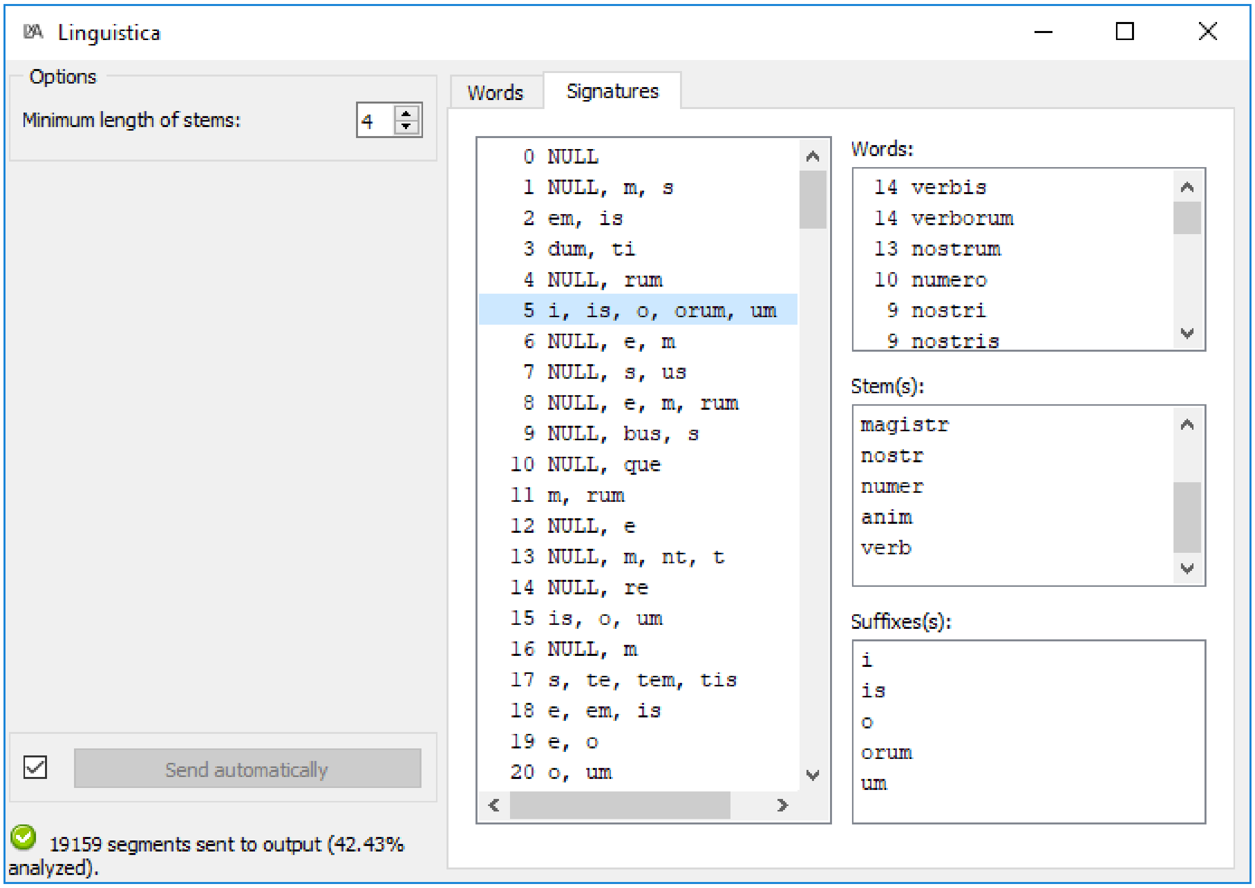
☑ Send automatically

19159 segments sent to output (42.43% analyzed).

Fig. 9: Figure 1: **Linguistica** widget interface (**Words** tab).

The right hand side of the interface contains two tabs that can be used to explore the discovered morphology. The **Words** tab displays the list of input words ordered by decreasing frequency. When a word is selected in this list, the possible parses are displayed in the **Parse(s)** section, along with a probability estimate (NB: at this point the estimation is excessively biased toward parses involving the "NULL" suffix, i.e. the empty string). When a parse is selected, the stems and suffixes that belong to the corresponding signature are displayed in the **Stem(s)** and **Suffixe(s)** lists. The signature's identifier (an integer) is also indicated.

The **Signatures** tab displays the list of discovered signatures, along with their identifier. When a signature is selected, the corresponding words, stems and suffixes are displayed in the other lists.



Minimum length of stems: 4

```
 0 NULL
 1 NULL, m, s
 2 em, is
 3 dum, ti
 4 NULL, rum
 5 i, is, o, orum, um
 6 NULL, e, m
 7 NULL, s, us
 8 NULL, e, m, rum
 9 NULL, bus, s
10 NULL, que
11 m, rum
12 NULL, e
13 NULL, m, nt, t
14 NULL, re
15 is, o, um
16 NULL, m
17 s, te, tem, tis
18 e, em, is
19 e, o
20 o, um
```

Words:

```
14 verbis
14 verborum
13 nostrum
10 numero
 9 nostri
 9 nostris
```

Stem(s):

```
magistr
nostr
numer
anim
verb
```

Suffixes(s):

```
i
is
o
orum
um
```

☑ 19159 segments sent to output (42.43% analyzed).

Fig. 10: Figure 2: **Linguistica** widget interface (**Signatures** tab).

The **Info** section indicates that the input has been correctly processed, or the reason why no output is emitted (no input, etc.). It also indicates the proportion of tokens that have been assigned to a signature (with the exception of the signature #0, which always contains all the stems that have only been found to occur with the NULL suffix).

The **Send** button triggers the computation and emission of the annotated word segmentation. When it is selected, the checkbox to the left of the button disables the button and the widget attempts to automatically emit results at every modification of its interface.

1.8.4 Messages

Information

<n> segments sent to output (<p>% analyzed). This confirms that the widget has operated properly.

Warnings

Unable to find any stems in data. The morphology learning algorithm has not been able to discover any relevant structure.

Unable to find any stems in data. Please check that they are segmented into words. The morphology learning algorithm has not been able to discover any relevant structure, and a likely explanation is that the input has not been segmented before being transmitted to this widget. If so, please segment it into words (for example using Textable's **Segment** widget).

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Widget needs input A Textable segmentation containing words should be input in the widget.

1.9 spaCy

Natural language processing using the spaCy (https://spacy.io/) library.

1.9.1 Author

Aris Xanthos

1.9.2 Signals

Inputs: `Text data`

Textable segmentation

Outputs:

- `Tokenized text` (default)

  Segmentation with a segment for each token in the input data

- `Named entities` (optional)

  Segmentation with a segment for each named entity in the input data

- `Noun chunks` (optional)

  Segmentation with a segment for each noun chunk in the input data

- `Sentences` (optional)

  Segmentation with a segment for each sentence in the input data

1.9.3 Description

This widget provides a graphical interface to a number of functionalities of the spaCy (https://spacy.io/) natural language processing Python library:

- tokenization
- part-of-speech tagging
- lemmatization
- named entity recognition
- noun chunk segmentation
- sentence segmentation

The user is referred to the extensive documentation of spaCy for detailed explanations of what these various levels of linguistic analysis encompass and how they are technically obtained. Note that spaCy is able to process text in a range of languages (cf. https://spacy.io/usage/models#languages), provided that the corresponding language "models" have been downloaded by the user, a task that this widget can do for you.

The widget outputs at least one segmentation containing a segment for each token in the input data. Segments in this segmentation have a variable number of annotations (depending on user-defined parameters and what is available for the language in question). For example, here is what spaCy's *en_core_web_sm* model returns for token *library* in the sentence *This library rocks.* (see spaCy's documentation for details):

key	example value
*dep_*	*nsubj*
*ent_iob_*	*O*
head	*rocks*
is_alpha	*True*
is_bracket	*False*
is_digit	*False*
is_left_punct	*False*
is_lower	*True*
is_oov	*True*
is_punct	*False*
is_quote	*False*
is_right_punct	*False*
is_space	*False*
is_stop	*False*
is_title	*False*
is_upper	*False*
*lang_*	*en*
*lemma_*	*library*
like_email	*False*
like_num	*False*
like_url	*False*
*lower_*	*library*
*norm_*	*library*
*pos_*	*NOUN*
sentiment	*0.0*
*shape_*	*xxxx*
*tag_*	*NN*
*whitespace_*	*" "*

Optionally, the widget's output may also include up to three more segmentations, into named entities, noun chunks <https://spacy.io/usage/linguistic-features#noun-chunks>'_, and sentences. These elements have the annotations *lemma_*, *lower_* and *sentiment*, as well as *label* (for all but sentences).

Interface

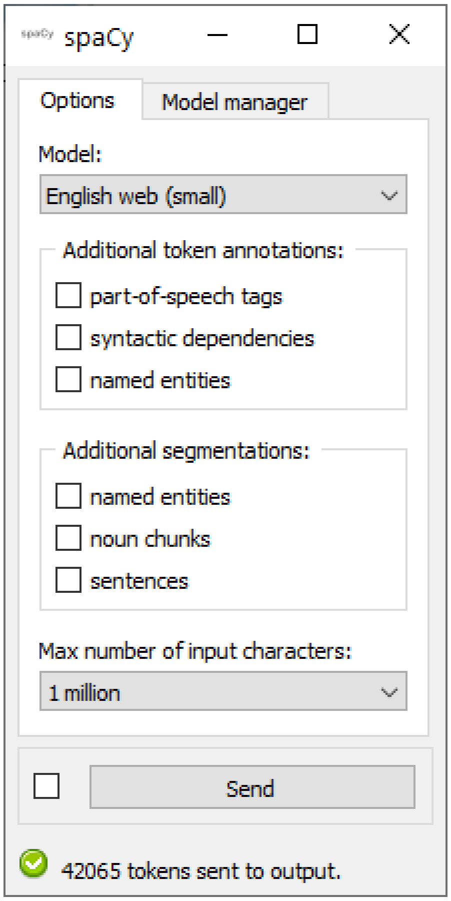
User controls are divided into two tabs: (see *figure 1* below): **Options** and **Model manager**.

Options tab

The **Options** tab contains all controls related to the way spaCy processes the input data.

The **Model** dropdown menu lets the user specify the language model to be used, among those that have been installed on their computer (see below for how to download and install models using the **Model manager** tab).

Regardless of any configuration choices, a given language model will at least output a tokenized version of the input data, with a subset of the annotations indicated above. By ticking boxes in the **Additional token annotations** section, the user can opt to add information concerning **part-of-speech tags**, **syntactic dependencies**, and **named entities**. Note that ticking these boxes may require to reload the language model (which can take some time, depending on model size), and will increase the duration of processing (in proportion of the amount of input data).



Model: English web (small)

Additional token annotations:

- □ part-of-speech tags
- □ syntactic dependencies
- □ named entities

Additional segmentations:

- □ named entities
- □ noun chunks
- □ sentences

Max number of input characters: 1 million

□ Send

✓ 42065 tokens sent to output.

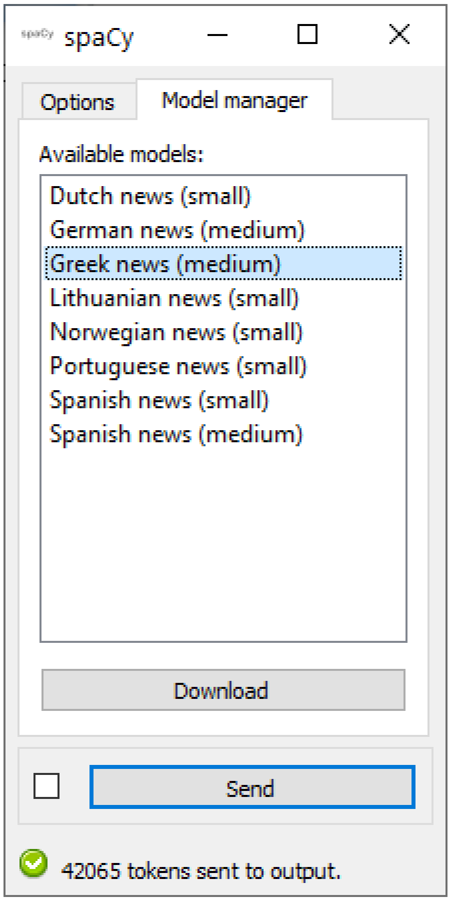
Fig. 11: Figure 1: **spaCy** widget interface, **Options** tab.

When boxes in the **Additional segmentations** are ticked, the widget will send up to three additional segmentations on separate output channels (which can be accessed by double-clicking the connexions between the **spaCy** widget and the next widget in the line and redrawing the connexions as desired in the **Edit Links** dialog). The segments of these segmentations correspond to **named entities**, **noun chunks**, and **sentences** respectively. The same remarks as for additional annotations apply: ticking these boxes may require to reload the language mode and will increase the duration of processing.

The last item in the **Options** section controls the **maximum number of input characters** allowed by the widget. As indicated in spaCy's documentation, the spaCy parser and NER models require roughly 1GB of temporary memory per 100‘000 characters in the input; this means long texts may cause memory allocation errors. It is probably safe to increase the default limit of 1 million characters if you're not using the syntactic parser (required for syntactic dependency annotation as well as noun chunk and sentence segmentation) or named entity recognizer, or have a large amount of RAM available.

Model manager tab

The **spaCy** is initially installed with a single language model for English. The **Model manager** tab (see *figure 2* below) enables the user to download and install additional language models for English or for other languages (cf. https://spacy.io/usage/models#languages for available language models)



Available models:

- Dutch news (small)
- German news (medium)
- Greek news (medium)
- Lithuanian news (small)
- Norwegian news (small)
- Portuguese news (small)
- Spanish news (small)
- Spanish news (medium)

42065 tokens sent to output.

Fig. 12: Figure 2: **spaCy** widget interface, **Model Manager** tab.

Simply select one ore more models to download and install, then click **Download** and confirm your choices with **OK**. After the models have been downloaded and installed, you will be prompted to quit and restart Orange Canvas for changes to take effect. Please note that some models may be quite large and take a substantial amount of time to download (in particular the *en_core_web_lg* English model, which weighs 798 Mb).

1.9.4 Messages

Information

<n> tokens, <m> noun chunks, <l> entities and <k> sentences sent to output. This confirms that the widget has operated properly.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Widget needs input. The widget instance needs data to be sent to its input channel in order to process it.

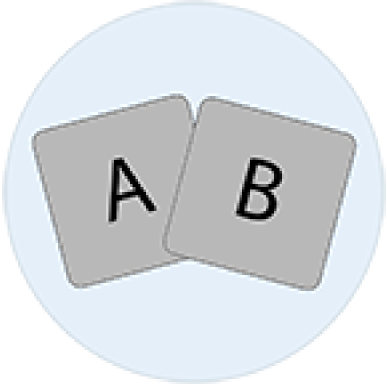
Please download a language model first. At least one language model needs to be installed before the widget can operate.

Loading language model, please wait… A language model is currently being downloaded and installed.

Processing, please wait… The requested NLP analysis is being performed.

Input exceeds max number of characters set by user. The number of characters in the widget's input is larger than the maximum number of characters allowed based user-defined settings; either decrease the input size or increase the maximum number of characters allowed.

1.10 Lexical Hunter



Annotates a segmentation based on one or more lexical fields.

1.10.1 Authors

- Maxime Bony
- Simon Cappelle
- Robin Pitteloud

1.10.2 Signals

Inputs:

- `Text segmentation`

Outputs:

- `Text segmentation`

1.10.3 Description

This widget is a lexical fields identifier. It creates annotation based on one or multiple lexical fields. There are some basic lexical fields, but the user can modify or remove them. The user can also create and/or import custom lexical fields. t is possible to export these lexical fields as well. The interface of Lexical Hunter is available in two versions, a basic one that lets the user select some lexical fields to annotate the segmented data and an advanced one that lets the user manage these lexical lists. Do not put any special characters in your lists, unless you know what you are doing, otherwise the widget will interpret them as a regex.

Basic interface

In its basic version (see *figure 1* below), the **Lexical Hunter** widget simply lets the user select one or more lexical fields in the default list. To select multiple files use either control/command-click or shift-click.

The **Annotation key** section allows the user to define the name of the annotation key.

The **Info** section indicates the number of segments in the output segmentation, or the reasons why no segmentation is emitted (segmentation input is needed, please select one or more lexical lists, etc.).

The **Send** button triggers the emission of a segmentation to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit a segmentation at every modification of its interface.

Advanced interface

The advanced version of **Lexical Hunter** (see *figure 2* below) lets the user edit, delete, add, import or export lists.

The **Edit** button puts the content of the selected list in the text field.

The **Import** button allows the user to import a list from a .txt file.

The **Export all** button allows the user to export every lexical field from the list. They are saved as .txt files.

The **Export selected** button allows the user to export the selected lexical field.

The **New** button allows the user to create a new lexical field.

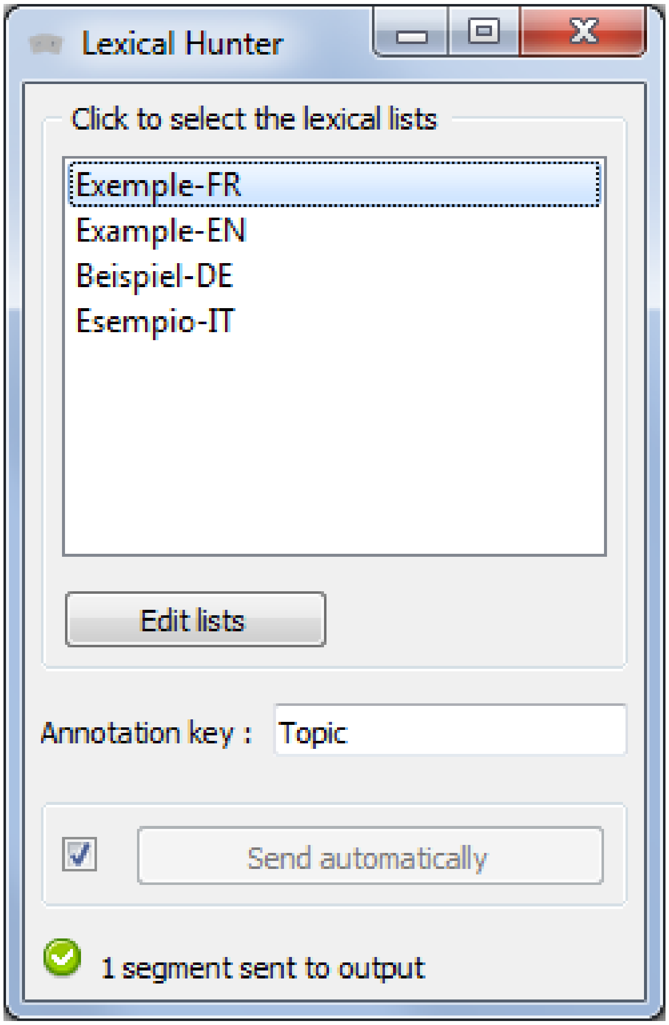
The **Clear all** button allows the user to delete all of the lexical lists.

The **Remove selected** button allows the user to delete the selected lexical list.

The **Commit** button saves the changes made to the list that is in the text field.

The **Save changes** button saves all the changes made by the user to the lists.

The **Cancel button** cancels all the changes made by the user to the lists.

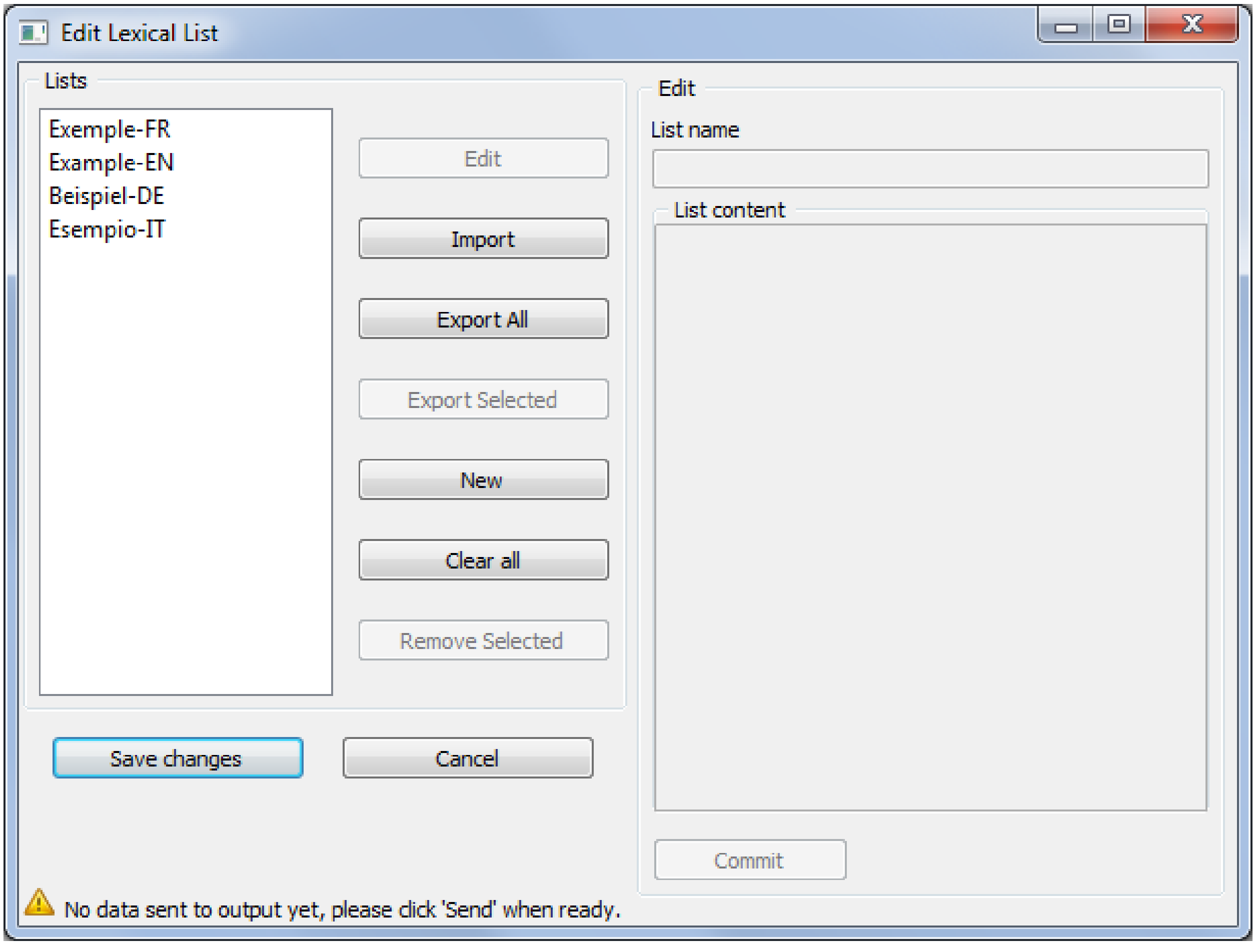


Annotation key : Topic

☑ Send automatically

1 segment sent to output

Fig. 13: Figure 1: **Lexical Hunter** widget (basic interface).



⚠ No data sent to output yet, please click 'Send' when ready.

Fig. 14: Figure 2: **Lexical Hunter** widget (advanced interface).

1.10.4 Messages

Information

<n> segments sent to output. This confirms that the widget has operated properly.

Warnings

A segmentation input is needed. The widget needs an input segmentation to process data.

An annoatation key is needed. The widget needs an annotation key value.

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Please select one or more lexical lists. The widget instance is not able to emit data to output because no lexical list has been selected.

No data sent to output yet: no input segmentation. The widget instance is not able to emit data to output because it receives none on its input channel(s).

Errors

You need to define at least one lexical list The user needs to define at least one lexical list.

Note

The widget still has a problem with the edit window on Windows. You have to close it twice or click twice the "Save changes" or "Cancel changes" button to close it.

1.11 Topic Models

Build topic models based on term-document matrices.

1.11.1 Author

Aris Xanthos

1.11.2 Signals

Input:

- `Textable crosstab`

  A term-document matrix in Textable PivotCrosstab format

Outputs:

- `Term-topic Textable table`

  A table (in Textable PivotCrosstab format) showing the association between terms and topics

- `Document-topic Textable table`

  A table (in Textable PivotCrosstab format) showing the association between documents and topics

- `Term-topic Orange table`

  A table (in Orange format) showing the association between terms and topics

- `Document-topic Orange table`

  A table (in Orange PivotCrosstab format) showing the association between documents and topics

1.11.3 Description

This widget takes a term-document matrix in input (such as emitted by Textable's **Count** widget) and applies one of several topic modelling methods to these data in order to infer latent, fuzzy word and document categories.

Two of the underlying methods (Latent Dirichlet and Latent semantic indexing allocation) are based on the Gensim third-party package while the third method (correspondence analysis) uses Orange's internal implementation.

The widget's output are two pairs of tables (one in Textable format and one in Orange format): term-topic tables show how strongly each topic is associated to each term, and document-topic tables displays their association with each document.

In addition, the widget's interface shows the list of terms that are most strongly associated with each topic. In the case of Latent semantic indexing and Correspondence analysis, the displayed terms are those that are either positively or negatively associated with each latent dimension (or factor, or component), and an indication of the proportion of variance (or inertia) explained by each topic is also given (see *figure 1* below).

Interface

The widget's interface requires little input from the user (see *figure 1* below): the desired topic modelling **Method** (Latent Dirichlet allocation, Latent semantic indexing, or Correspondence analysis) and the **Number of topics** to be computed.

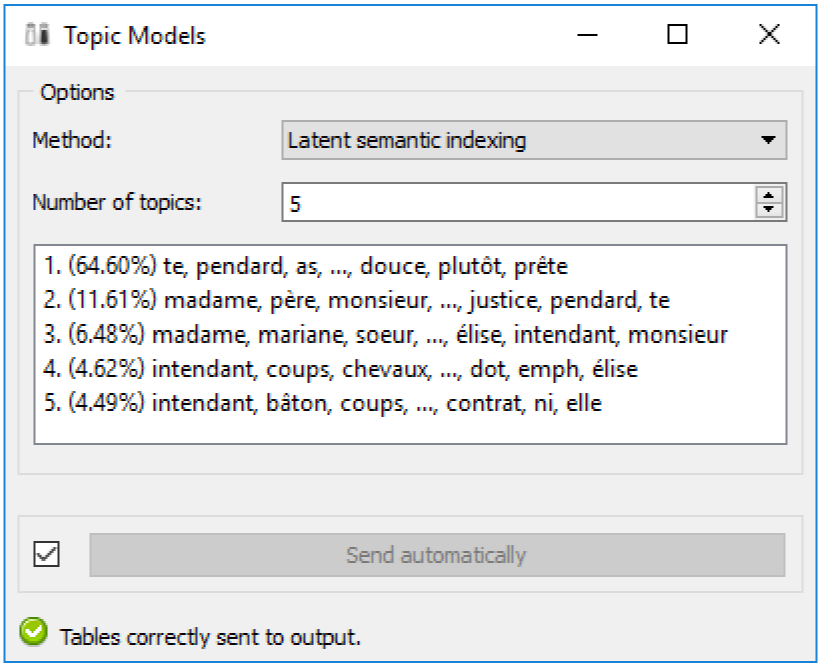
The **Info** section indicates that the input has been correctly processed, or the reason why no output is emitted (no input, etc.).

The **Send** button triggers the computation and emission of term-topic and document-topic tables to the output connection(s). When it is selected, the **Send automatically** checkbox disables the button and the widget attempts to automatically emit results at every modification of its interface.

1.11.4 Messages

Information

Tables correctly sent to output. This confirms that the widget has operated properly.



Method: Latent semantic indexing

Number of topics: 5

1. (64.60%) te, pendard, as, ..., douce, plutôt, prête
2. (11.61%) madame, père, monsieur, ..., justice, pendard, te
3. (6.48%) madame, mariane, soeur, ..., élise, intendant, monsieur
4. (4.62%) intendant, coups, chevaux, ..., dot, emph, élise
5. (4.49%) intendant, bâton, coups, ..., contrat, ni, elle

☑ Send automatically

Tables correctly sent to output.

Fig. 15: Figure 1: **Topic Models** widget interface.

Warnings

Settings were changed, please click 'Send' when ready. Settings have changed but the **Send automatically** checkbox has not been selected, so the user is prompted to click the **Send** button (or equivalently check the box) in order for computation and data emission to proceed.

Widget needs input A term-document matrix (in Textable PivotCrosstab format) should be input in the widget.