

# Package: MetaNLP (via r-universe)

December 8, 2024

**Type** Package

**Title** Natural Language Processing for Meta Analysis

**Version** 0.1.3

**Description** Given a CSV file with titles and abstracts, the package creates a document-term matrix that is lemmatized and stemmed and can directly be used to train machine learning methods for automatic title-abstract screening in the preparation of a meta analysis.

**License** MIT + file LICENSE

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0), covr, wordcloud, vdiff

**Imports** glmnet, tm, textstem, methods, lexicon, utils

**Collate** MetaNLP.R util.R delete\_functions.R feature\_selection.R useful\_functions.R

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**BugReports** <https://github.com/imbi-heidelberg/MetaNLP/issues>

**URL** <https://github.com/imbi-heidelberg/MetaNLP>

**Config/testthat/edition** 3

**Depends** R (>= 2.10)

**NeedsCompilation** no

**Author** Nico Bruder [aut] (<<https://orcid.org/0009-0004-9522-2075>>), Samuel Zimmermann [aut] (<<https://orcid.org/0009-0000-4828-9294>>), Johannes Vey [aut] (<<https://orcid.org/0000-0002-2610-9667>>), Maximilian Pilz [aut, cre] (<<https://orcid.org/0000-0002-9685-1613>>), Institute of Medical Biometry - University of Heidelberg [cph]

**Maintainer** Maximilian Pilz <[maximilian.pilz@itwm.fraunhofer.de](mailto:maximilian.pilz@itwm.fraunhofer.de)>

**Repository** CRAN

**Date/Publication** 2024-11-07 13:30:02 UTC

**Config/pak/sysreqs** libicu-dev libxml2-dev

## Contents

delete_stop_words . . . . .	2
delete_words . . . . .	3
MetaNLP . . . . .	4
plot,MetaNLP,missing-method . . . . .	5
read_test_data . . . . .	6
replace_special_characters . . . . .	7
select_features . . . . .	8
summary,MetaNLP-method . . . . .	9
write_csv . . . . .	10

<b>Index</b>	<b>12</b>
--------------	-----------

---

delete_stop_words	<i>Delete stop words</i>
-------------------	--------------------------

---

### Description

Usually, stop words do not offer useful information in the classification whether a paper should be included or excluded from a meta-analysis. Thus, such words should not be part of the document-term matrix. This function allows the user to automatically delete stop words.

### Usage

```
delete_stop_words(object, ...)

## S4 method for signature 'MetaNLP'
delete_stop_words(object, ...)
```

### Arguments

object	A MetaNLP object, whose data frame is to be modified.
...	Language of the stop words. Defaults to "english".

### Details

This function allows to delete stop words from different languages. Supported languages are english, french, german, russian and spanish. Language names are case sensitive.

### Value

An object of class MetaNLP.

### Examples

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
obj <- delete_stop_words(obj, "english")
```

---

delete_words	<i>Delete list of words</i>
--------------	-----------------------------

---

### Description

There can be words that do not offer additional information in the classification whether a paper should be included or excluded from a meta-analysis. Thus, such words should not be part of the document-term matrix. This function allows the user to remove these columns of the word count matrix by specifying a vector of words to delete.

### Usage

```
delete_words(object, delete_list)

## S4 method for signature 'MetaNLP,character'
delete_words(object, delete_list)
```

### Arguments

object	A MetaNLP object, whose data frame is to be modified
delete_list	A character vector containing the words to be deleted

### Details

The words in `delete_list` can be given like they appear in the text. They are lemmatized and stemmed by `delete_words` to match the columns of the document-term matrix.

### Value

An object of class MetaNLP

### Examples

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
del_words <- c("beautiful", "considering", "found")
obj <- delete_words(obj, del_words)
```

## Description

The **MetaNLP** package provides methods to quickly transform a CSV-file with titles and abstracts to an R data frame that can be used for automatic title-abstract screening using machine learning.

A MetaNLP object is the base class of the package **MetaNLP**. It is initialized by passing the path to a CSV file and constructs a data frame whose column names are the words that occur in the titles and abstracts and whose cells contain the word frequencies for each paper.

## Usage

```
MetaNLP(
  file,
  bounds = c(2, Inf),
  word_length = c(3, Inf),
  language = "english",
  ...
)
```

## Arguments

file	Either the path to the CSV file or a data frame containing the abstracts
bounds	An integer vector of length 2. The first value specifies the minimum number of appearances of a word to become a column of the word count matrix, the second value specifies the maximum number. Defaults to <code>c(2, Inf)</code> .
word_length	An integer vector of length 2. The first value specifies the minimum number of characters of a word to become a column of the word count matrix, the second value specifies the maximum number. Defaults to <code>c(3, Inf)</code> .
language	The language for lemmatization and stemming. Supported languages are <code>english</code> , <code>french</code> , <code>german</code> , <code>russian</code> and <code>spanish</code> . For non-english languages make sure that the csv which is processed has the correct encoding.
...	Additional arguments passed on to <code>read.csv2</code> , e.g. when <code>"</code> should be used as a separator or when the encoding should be changed. See <a href="#">read.table</a> .

## Details

An object of class MetaNLP contains a slot `data_frame` where the document-term matrix is stored as a data frame. The CSV file must have a column `ID` to identify each paper, a column `title` with the belonging titles of the papers and a column `abstract` which contains the abstracts. If the CSV stores training data, a column `decision` should exist, indicating whether an abstract is included in the meta analysis. This column does not need to exist, because there is no decision for test data yet. Allowed values in this column are either `"yes"` and `"no"` or `"include"` and `"exclude"` or `"maybe"`. The value `"maybe"` is handled as a `"yes"/"include"`.

**Value**

An object of class MetaNLP

**Note**

To ensure correct processing of the data when there are special characters (e.g. "é" or "ü"), make sure that the csv-file is correctly encoded as UTF-8. The stemming algorithm makes use of the C libstemmer library generated by Snowball. When german texts are stemmed, umlauts are replaced by their non-umlaut equivalent, so "ä" becomes "a" etc.

**Author(s)**

**Maintainer:** Maximilian Pilz <maximilian.pilz@itwm.fraunhofer.de> ([ORCID](#))

Authors:

- Nico Bruder <brudernico@gmail.com> ([ORCID](#))
- Samuel Zimmermann <zimmermann@imbi.uni-heidelberg.de> ([ORCID](#))
- Johannes Vey <vey@imbi.uni-heidelberg.de> ([ORCID](#))

Other contributors:

- Institute of Medical Biometry - University of Heidelberg [copyright holder]

**See Also**

Useful links:

- <https://github.com/imbi-heidelberg/MetaNLP>
- Report bugs at <https://github.com/imbi-heidelberg/MetaNLP/issues>

**Examples**

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
```

---

plot,MetaNLP,missing-method

*Create word cloud from MetaNLP-object*

---

**Description**

This method creates a word cloud from a MetaNLP object. The word size indicates the frequency of the words.

**Usage**

```
## S4 method for signature 'MetaNLP,missing'
plot(
  x,
  y = NULL,
  max.words = 70,
  colors = c("snow4", "darkgoldenrod1", "turquoise4", "tomato"),
  decision = c("total", "include", "exclude"),
  ...
)
```

**Arguments**

x	A MetaNLP object to plot
y	not used
max.words	Maximum number of words in the word cloud
colors	Character vector with the colors in
decision	Stratify word cloud by decision. Default is no stratification.
...	Additional parameters for <a href="#">wordcloud</a>

**Value**

nothing

**Examples**

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
plt <- plot(obj)
```

---

read\_test\_data

*Read and adapt test data*


---

**Description**

This function takes a MetaNLP object (the training data) and the test data. The function creates the document-term matrix from the test data and matches the columns of the given training MetaNLP object with the columns of the test document-term matrix. This means that columns, which do appear in the test document-term matrix but not in the training document-term matrix are removed; columns that appear in the training document-term matrix but not in the test document-term matrix are added as a column consisting of zeros.

**Usage**

```
read_test_data(object, ...)  
  
## S4 method for signature 'MetaNLP'  
read_test_data(object, file, ...)
```

**Arguments**

object	The MetaNLP object created from the training data.
...	Further arguments to MetaNLP.
file	Either the path to the test data csv, the data frame containing the papers or a MetaNLP object

**Value**

An object of class MetaNLP

**Examples**

```
path_train <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)  
path_test <- system.file("extdata", "test_data_changed.csv", package = "MetaNLP", mustWork = TRUE)  
obj_train <- MetaNLP(path_train)  
obj_test <- MetaNLP(path_test)  
to_test_obj <- read_test_data(obj_train, obj_test)
```

---

replace\_special\_characters

*Replace special characters in column names*

---

**Description**

When using non-english languages, the column names of the document-term matrix can contain special characters. These might lead to encoding problems, when this matrix is used to train a machine learning model. This functions automatically replaces all special characters by the nearest equivalent character, e.g. "é" would be replaced by "e".

**Usage**

```
replace_special_characters(object)  
  
## S4 method for signature 'MetaNLP'  
replace_special_characters(object)
```

**Arguments**

object	An object of class MetaNLP.
--------	-----------------------------

**Value**

An object of class MetaNLP, where the column names do not have special characters anymore.

**Examples**

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path, language = "french")
obj <- replace_special_characters(obj)
```

---

select_features	<i>Select features via elasticnet regularization</i>
-----------------	------------------------------------------------------

---

**Description**

As the document-term matrix quickly grows with an increasing number of abstracts, it can easily reach several thousand columns. Thus, it can be important to extract the columns that carry most of the information in the decision making process. This function uses a generalized linear model combined with elasticnet regularization to extract these features. In contrast to a usual regression model or a L2 penalty (ridge regression), elasticnet (and LASSO) sets some regression parameters to 0. Thus, the selected features are exactly the features with a non-zero entry.

**Usage**

```
select_features(object, ...)

## S4 method for signature 'MetaNLP'
select_features(object, alpha = 0.8, lambda = "avg", seed = NULL, ...)
```

**Arguments**

object	An object of class MetaNLP
...	Additional arguments for <a href="#">cv.glmnet</a> . An important option might be <code>type.measure</code> to specify which loss is used when the cross validation is executed.
alpha	The elastic net mixing parameter, with $0 \leq \alpha \leq 1$ . <code>alpha = 1</code> then equals the lasso penalty, <code>alpha = 0</code> is the ridge penalty.
lambda	The weight parameter of the penalty. The possible values are "avg", "min", "1se" or a numeric value which directly determines $\lambda$ . When choosing "avg", "min" or "1se", cross validation is executed to determine $\lambda$ . Note that cross validation uses random folds, so the results are not necessarily replicable. "avg" calls <code>select_features</code> 10 times, computes the $\lambda$ which minimizes the loss for each iteration and then uses the median of these values as the final value, for which the objective function is minimized. "min" and "1se" carry out the cross validation just once and $\lambda$ is either the value, for which the cross-validated error is minimized (option "min") or the value, that gives the most regularized model such that the cross-validated error is within one standard error of the minimum (option "1se").



**seed** A numeric value which is used as a local seed for this function. Default is seed = NULL, so no seed is set. Setting a seed leads to replicable results of the cross validation, such that each call of `select_features` selects the same columns. If a seed is set, the option `lambda = "avg"` yields the same results as `lambda = "min"`.

### Details

The computational aspects are executed by the `glmnet` package. At first, a model is fitted via [glmnet](#). The elastic net parameter  $\alpha$  can be specified by the user. The parameter  $\lambda$ , which determines the weight of the penalty, can either be chosen via cross validation (using [cv.glmnet](#) or by giving a numeric value.

### Value

An object of class `MetaNLP`, where the columns were selected via elastic net.

### Note

By using a fix value for `lambda`, the number of features which should be selected can easily be adjusted by the parameter `alpha`. The smaller one chooses `alpha`, the more columns will still be present in the resulting data frame, the higher one chooses `alpha`, the less columns will be chosen.

### Examples

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
obj2 <- select_features(obj, alpha = 0.7, lambda = "min")
```

---

summary,MetaNLP-method

*Summary of MetaNLP-objects*

---

### Description

Returns a quick overview over the  $n$  most frequent word stems structured into included and excluded papers.

### Usage

```
## S4 method for signature 'MetaNLP'
summary(object, n = 5, stop_words = FALSE, ...)
```

**Arguments**

object	An object of class MetaNLP.
n	Number of most frequent words to be displayed.
stop_words	Boolean to decide whether stop words shall be included in the summary. stop_words = TRUE means, that stop words are included.
...	Additional parameters for delete_stop_words (e.g. language of the stop words).

**Value**

A list of most frequent words.

**Examples**

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
summary(obj, n = 8)
```

---

write\_csv

*Save the document-term matrix*

---

**Description**

This function can be used to save the document-term matrix of a MetaNLP object as a csv-file.

**Usage**

```
write_csv(object, ...)

## S4 method for signature 'MetaNLP'
write_csv(object, path, type = c("train", "test"), ...)
```

**Arguments**

object	An object of class MetaNLP.
...	Additional arguments for <a href="#">write.table</a> , e.g. encoding as UTF-8.
path	Path where to save the csv.
type	Specifies if the document-term matrix should be saved as "train_wcm.csv" or "test_wcm.csv". If the user wants to use another file name, the whole path including the file name should be given as the path argument

**Details**

If a path to a specific folder is given (but the path name does not end with ".csv"), the file is saved in this folder as "train\_wcm.csv" or "test\_wcm.csv". By providing a path ending with ".csv", the user can override the default naming convention and the file is saved according to this path.

**Value**

nothing

**Examples**

```
path <- system.file("extdata", "test_data.csv", package = "MetaNLP", mustWork = TRUE)
obj <- MetaNLP(path)
obj2 <- delete_stop_words(obj)
write_path <- tempdir()
write_csv(obj2, path = write_path)
file.remove(file.path(write_path, "train_wcm.csv"))
```

# Index

`cv.glmnet`, [8](#), [9](#)

`delete_stop_words`, [2](#)

`delete_stop_words`, MetaNLP-method  
(`delete_stop_words`), [2](#)

`delete_words`, [3](#)

`delete_words`, MetaNLP, character-method  
(`delete_words`), [3](#)

`glmnet`, [9](#)

MetaNLP, [4](#)

MetaNLP-class (MetaNLP), [4](#)

`plot`, MetaNLP, missing-method, [5](#)

`read.table`, [4](#)

`read_test_data`, [6](#)

`read_test_data`, MetaNLP-method  
(`read_test_data`), [6](#)

`replace_special_characters`, [7](#)

`replace_special_characters`, MetaNLP-method  
(`replace_special_characters`), [7](#)

`select_features`, [8](#)

`select_features`, MetaNLP-method  
(`select_features`), [8](#)

`summary`, MetaNLP-method, [9](#)

`wordcloud`, [6](#)

`write.table`, [10](#)

`write_csv`, [10](#)

`write_csv`, MetaNLP-method (`write_csv`), [10](#)