Package 'RmecabKo'

July 15, 2026

Type Package

Title Korean Text Analysis with 'MeCab'

Version 0.3.0

Description A Korean text-analysis layer over the 'MeCab' morphological analyzer. Provides tokenizers that follow the 'tokenizers' contract for use with 'tidytext', morpheme-aware n-grams, a curated Korean stopword table, access to the KNU sentiment lexicon, friendly user-dictionary management, predicate lemmatization, keyword extraction, keyword-in-context concordances, and light text normalization. The native 'MeCab' interface and dictionary compilation are provided by 'RcppMeCab'.

Depends R (>= 3.6.0)

Imports Rcpp, RcppMeCab (>= 0.0.1.7), stringi

LinkingTo Rcpp

Suggests testthat (>= 3.0.0), withr, jsonlite, tibble, dplyr, tidytext, knitr, rmarkdown

License GPL (>= 2)

URL https://github.com/junhewk/RmecabKo

BugReports https://github.com/junhewk/RmecabKo/issues

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Junhewk Kim [aut, cre]

Maintainer Junhewk Kim <junhewk.kim@gmail.com>

Repository CRAN

Date/Publication 2026-07-15 07:30:02 UTC

Contents

demo_ko . . . 2
dict_add_words . . . 3
install_mecab . . . 4
keywords_textrank . . . 5
keywords_tfidf . . . 6
kwic . . . 7
lemmatize_morphemes . . . 8
lexicon_knu . . . 9
nouns . . . 10
pos . . . 11
stopwords_ko . . . 11
stopwords_ko_tags . . . 12
stopwords_ko_words . . . 13
text_normalize . . . 13
token_lemma . . . 14
token_morph . . . 15
token_ngrams . . . 17
words . . . 18

Index **19**

`demo_ko` *Korean demonstration sentences*

Description

A small public-domain corpus of Korean sentences used in the package examples and vignette. The text is short, self-contained, and free of any third-party licensing so it can ship on CRAN.

Usage

```
demo_ko
```

Format

A named character vector of Korean sentences.

See Also

‘vignette("korean-text-analysis", package = "RmecabKo")‘

dict_add_words *Manage a MeCab user dictionary from R*

Description

These functions provide a friendly layer over [RcppMeCab::dict_index()] for teaching the analyzer new words - proper nouns, neologisms, domain terms - without hand-writing ‘mecab-ko-dic‘ CSV rows. Words are kept in a named registry in the user data directory, compiled to a binary dictionary, and activated for the current session.

Usage

```
dict_add_words(
  words,
  tag = "NNP",
  reading = NULL,
  meaning = NULL,
  cost = 3000L,
  name = "user",
  sys_dic = "",
  compile = TRUE
)

dict_words(name = "user")

dict_remove_words(words, name = "user", compile = TRUE)

dict_compile(name = "user", sys_dic = "")

dict_use(name = "user")

dict_path(name = "user")
```

Arguments

words	Either a character vector of surface forms or a data frame with a ‘surface‘ column and optional ‘tag‘, ‘reading‘, ‘meaning‘, ‘cost‘, ‘left_id‘, and ‘right_id‘ columns. For ‘dict_remove_words()‘, a character vector of surfaces to drop.
tag	Default ‘mecab-ko-dic‘ POS tag for character-vector input, typically ‘"NNP"‘ (proper noun) or ‘"NNG"‘ (common noun).
reading	Optional reading; defaults to the surface form.
meaning	Optional semantic class (the ‘mecab-ko-dic‘ semantic field, for example a personal-name or place-name class); defaults to ‘"*"‘.
cost	Integer cost. Lower values make the word more likely to be chosen during analysis.

name	Dictionary name, or an absolute path to a dictionary directory for per-project dictionaries.
sys_dic	Optional Korean system dictionary directory. Defaults to the active dictionary.
compile	Recompile the binary dictionary after the change.

Details

'dict_add_words()' appends words and (by default) recompiles. 'dict_words()' returns the current registry, 'dict_remove_words()' drops entries, 'dict_compile()' rebuilds the binary dictionary, 'dict_use()' activates it for later calls to [pos()] and the tokenizers, and 'dict_path()' returns the path of the compiled dictionary for use with the 'user_dic' argument.

The left/right context IDs and jongseong (final-consonant) flag required by 'mecab-ko-dic' are filled in automatically from the active system dictionary's 'left-id.def' and 'right-id.def'. Supply 'left_id'/'right_id' columns to override them, and tune 'cost' (lower costs make a word more likely to be selected).

Value

'dict_add_words()', 'dict_remove_words()', and 'dict_compile()' return the registry data frame invisibly; 'dict_words()' returns it visibly. 'dict_use()' and 'dict_path()' return the compiled dictionary path.

Examples

```
## Not run:
dict_add_words(c("\uc740\uc804\ud55c\ub2e2", "\uce74\ube44\ubd07"), tag = "NNP")
dict_use()
pos("\uce74\ube44\ubd07 \ucd9c\uc2dc")
dict_words()
dict_remove_words("\uce74\ube44\ubd07")

## End(Not run)
```

---

install_mecab *Deprecated MeCab installer*

Description

'RcppMeCab' now installs the native MeCab engine and Korean dictionary. Install it with 'install.packages("RcppMeCab")' instead.

Usage

```
install_mecab(mecabLocation)
```

Arguments

mecabLocation	Ignored legacy installation path.

Value

Invisibly returns ‘NULL‘.

`keywords_textrank` *Keyword extraction with TextRank*

Description

Ranks morphemes within each document using the TextRank graph algorithm: tokens that co-occur within a sliding window vote for one another, and the stationary scores of the resulting graph surface the most central terms. Unlike [keywords_tfidf()], TextRank scores each document on its own and needs no reference corpus.

Usage

```
keywords_textrank(
  phrase,
  div = c("nouns", "words", "morph"),
  top_n = 10L,
  window = 2L,
  stopwords = character(),
  damping = 0.85,
  iter = 30L,
  sys_dic = "",
  user_dic = "",
  parallel = FALSE
)
```

Arguments

`phrase`	A character vector or list of character scalars.
`div`	Token preset: nouns, content words, or all morphemes.
`top_n`	Number of keywords to keep per document, ranked by TF-IDF. Use ‘Inf‘ for all terms.
`window`	Co-occurrence window width in tokens.
`stopwords`	Character tokens to drop before scoring. Combine with [stopwords_ko_words()].
`damping`	Damping factor for the random-walk model.
`iter`	Maximum number of power-iteration steps.
`sys_dic`	Optional Korean system dictionary directory.
`user_dic`	Optional compiled user dictionary.
`parallel`	Use parallel analysis for multiple documents.

Value

A data frame with columns ‘doc‘, ‘word‘, and ‘score‘, ordered by document and descending score.

See Also

[keywords_tfidf()]

Examples

```
## Not run:
text <- paste("\ud55c\uad6d\uc5b4 \ubd84\uc11d \ub3c4\uad6c\ub294",
              "\ud55c\uad6d\uc5b4 \ucc98\ub9ac\ub97c \ub3d5\ub294\ub2e4")
keywords_textrank(text)

## End(Not run)
```

`keywords_tfidf` *Keyword extraction with TF-IDF*

Description

Ranks the morphemes of each document by term frequency-inverse document frequency over the supplied corpus. Tokens are produced with one of the Korean presets, so keywords are morphologically normalized rather than raw whitespace tokens.

Usage

```
keywords_tfidf(
  phrase,
  div = c("nouns", "words", "morph"),
  top_n = 10L,
  stopwords = character(),
  sys_dic = "",
  user_dic = "",
  parallel = FALSE
)
```

Arguments

`phrase`	A character vector or list of character scalars.
`div`	Token preset: nouns, content words, or all morphemes.
`top_n`	Number of keywords to keep per document, ranked by TF-IDF. Use ‘Inf‘ for all terms.
`stopwords`	Character tokens to drop before scoring. Combine with [stopwords_ko_words()].
`sys_dic`	Optional Korean system dictionary directory.
`user_dic`	Optional compiled user dictionary.
`parallel`	Use parallel analysis for multiple documents.

Value

A data frame with columns ‘doc‘, ‘word‘, ‘n‘ (count), ‘tf‘, ‘idf‘, and ‘tf_idf‘, ordered by document and descending TF-IDF.

See Also

[keywords_textrank()]

Examples

```
## Not run:
docs <- c("\ud55c\uad6d\uc5b4 \ubd84\uc11d\uc740 \uc7ac\ubbf8\uc788\ub2e4",
          "\ubd84\uc11d \ub3c4\uad6c\uac00 \ud544\uc694\ud558\ub2e4")
keywords_tfidf(docs)

## End(Not run)
```

---

kwic *Keyword-in-context concordance*

Description

Finds occurrences of a keyword among the morphemes of each document and returns them with their left and right context, the Korean analogue of a classic KWIC concordance. Matching is done on morpheme tokens, so it is robust to the particles and endings that attach to a word in running text.

Usage

```
kwic(
  phrase,
  pattern,
  window = 5L,
  div = c("morph", "words", "nouns"),
  fixed = TRUE,
  sys_dic = "",
  user_dic = "",
  parallel = FALSE
)
```

Arguments

phrase	A character vector or list of character scalars.
pattern	A single search term. Matched exactly against a morpheme when ‘fixed = TRUE‘, or as a regular expression when ‘fixed = FALSE‘.
window	Number of context tokens to show on each side.
div	Token preset used to segment the text.

fixed	Match ‘pattern‘ exactly against a token; when ‘FALSE‘, treat it as a regular expression.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use parallel analysis for multiple documents.

Value

A data frame with columns ‘doc‘, ‘position‘ (token index of the match), ‘left‘, ‘keyword‘, and ‘right‘. Left and right context tokens are joined with single spaces.

Examples

```
## Not run:
docs <- c("\ud55c\uad6d\uc5b4 \ubd84\uc11d\uc740 \uc990\uac81\ub2e4",
          "\ud55c\uad6d\uc5b4 \uacf5\ubd80\ub294 \uc5b4\ub835\ub2e4")
kwic(docs, "\ud55c\uad6d\uc5b4")

## End(Not run)
```

---

lemmatize_morphemes *Recover dictionary forms of Korean predicates*

Description

Reconstructs the dictionary (citation) form of verbs and adjectives from MeCab morphemes by appending the terminal ending to a predicate stem, and by recombining a noun root with a following derivational suffix into a single predicate. This is useful for frequency counts and for matching against a sentiment lexicon, where inflected surfaces would otherwise scatter.

Usage

```
lemmatize_morphemes(tokens, combine_root = TRUE)
```

Arguments

tokens	A named character vector of morphemes with POS tags as names, as produced by ‘pos(x, join = FALSE)[[i]]‘.
combine_root	Recombine a noun root immediately before a derivational suffix (‘XSV‘/‘XSA‘) into a single predicate lemma.

Details

MeCab sometimes fuses a stem and its ending into a single token (tagged, for example, ‘VV+EP‘ or ‘VA+ETM‘). Because [RcppMeCab::pos()] does not expose the underlying morpheme decomposition, the original stem cannot be recovered for these fused tokens, and their ‘lemma‘ is returned as ‘NA‘.

Value

A data frame with columns ‘surface‘, ‘tag‘, and ‘lemma‘ (the dictionary form, or ‘NA‘ when it cannot be recovered).

See Also

[token_lemma()]

Examples

```
## Not run:
lemmatize_morphemes(pos("\uba39\uc5c8\ub2e4", join = FALSE)[[1]])

## End(Not run)
```

---

lexicon_knu *Korean sentiment lexicon (KNU)*

Description

Downloads and caches the KNU Korean sentiment lexicon (KnuSentiLex) and returns it as a tidy data frame suitable for joining against tokens produced by [token_morph()] or [token_nouns()]. The lexicon is not bundled with the package: on first use it is fetched from its public repository and stored in the user data directory, then read from that cache on later calls.

Usage

```
lexicon_knu(dir = NULL, force = FALSE, quiet = FALSE)
```

Arguments

dir	Directory for the cached copy. Defaults to a ‘lexicons‘ subfolder of ‘tools::R_user_dir("RmecabKo", "data")‘.
force	Re-download and overwrite the cached copy.
quiet	Suppress the download and attribution messages.

Details

The lexicon assigns each entry a polarity from ‘-2‘ (strongly negative) to ‘2‘ (strongly positive). Some entries are multi-word expressions; the ‘n_words‘ column gives the token count so you can restrict to single morphemes when joining.

The KNU sentiment lexicon (KnuSentiLex) is developed by researchers at Kyungpook National University (KNU) and is distributed under the Creative Commons Attribution-NonCommercial-ShareAlike (CC BY-NC-SA) license. It is therefore not bundled with this package, whose own license permits commercial use: ‘lexicon_knu()‘ downloads it from its source repository so that you obtain it directly and accept its terms. The **NonCommercial** clause restricts commercial use, and any redistribution must preserve attribution and the ShareAlike terms. Cite the lexicon when you use it and review the full terms at <https://github.com/park1200656/KnuSentiLex>.

Value

A data frame with columns ‘word‘ (UTF-8), ‘polarity‘ (integer ‘-2..2‘), and ‘n_words‘ (token count of ‘word‘).

Examples

```
## Not run:
senti <- lexicon_knu()
# keep single-morpheme entries only
senti[senti$n_words == 1L, ]

## End(Not run)
```

---

nouns *Extract Korean nouns*

Description

Keeps all ‘mecab-ko-dic‘ POS categories beginning with ‘N‘, including common and proper nouns, dependent nouns, numerals, and pronouns.

Usage

```
nouns(sentence, sys_dic = "", user_dic = "", parallel = FALSE)
```

Arguments

sentence	A character vector or list of character scalars.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use parallel analysis for multiple documents.

Value

A named list of character vectors.

pos *Korean part-of-speech tagging*

Description

Tags Korean text with the active ‘mecab-ko-dic‘ dictionary through [RcppMeCab::pos()].

Usage

```
pos(
  sentence,
  join = TRUE,
  format = c("list", "data.frame"),
  sys_dic = "",
  user_dic = "",
  parallel = FALSE
)
```

Arguments

sentence	A character vector or list of character scalars.
join	Whether to return ‘morpheme/POS‘ strings. When ‘FALSE‘, POS tags are stored as names on each morpheme vector.
format	Either ‘"list"‘ or ‘"data.frame"‘.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use [RcppMeCab::posParallel()] for multiple documents.

Value

A list for ‘format = "list"‘, or the data frame returned by ‘RcppMeCab‘.

stopwords_ko *Korean stopword table*

Description

A curated set of Korean function morphemes that are usually removed before content analysis: particles, endings, derivational suffixes, dependent nouns, and common function words. Each morpheme is tagged with its typical ‘mecab-ko-dic‘ part-of-speech tag so it can be filtered either by surface form or by tag.

Usage

```
stopwords_ko
```

Format

A data frame with one row per morpheme and three columns:

word Morpheme surface (UTF-8).

tag Typical ‘mecab-ko-dic‘ part-of-speech tag.

category Factor grouping the morpheme: ‘josa‘ (particles), ‘eomi‘ (endings), ‘suffix‘ (derivational suffixes), ‘formal_noun‘ (dependent nouns), or ‘function_word‘ (demonstratives, pronouns, conjunctive adverbs).

See Also

[stopwords_ko_words()], [stopwords_ko_tags()]

---

stopwords_ko_tags *Korean stopword POS tags*

Description

Returns the mecab-ko-dic part-of-speech tags in [stopwords_ko], optionally restricted to one or more categories. Pass the result to the ‘drop_pos‘ argument of [token_morph()] or [token_ngrams()] to strip whole classes of function morphemes at the tag level.

Usage

```
stopwords_ko_tags(category = NULL)
```

Arguments

`category`	Optional character vector of categories to keep: any of ‘"josa"‘, ‘"eomi"‘, ‘"suffix"‘, ‘"formal_noun"‘, ‘"function_word"‘. ‘NULL‘ returns every surface.

Value

A character vector of unique POS tags.

See Also

[stopwords_ko_words()], [stopwords_ko]

Examples

```
stopwords_ko_tags("josa")
```

stopwords_ko_words *Korean stopword surfaces*

Description

Returns the morpheme surfaces in [stopwords_ko], optionally restricted to one or more categories. Use the result for surface-level filtering, for example ‘dplyr::anti_join()‘ in a tidy pipeline or the ‘stopwords‘ argument of [token_ngrams()].

Usage

```
stopwords_ko_words(category = NULL)
```

Arguments

category	Optional character vector of categories to keep: any of ‘"josa"‘, ‘"eomi"‘, ‘"suffix"‘, ‘"formal_noun"‘, ‘"function_word"‘. ‘NULL‘ returns every surface.

Value

A character vector of unique morpheme surfaces.

See Also

[stopwords_ko_tags()], [stopwords_ko]

Examples

```
head(stopwords_ko_words())
stopwords_ko_words("josa")
```

text_normalize *Normalize Korean text before tokenizing*

Description

Applies light, rule-based cleanup that improves morphological analysis and downstream matching: Unicode NFC composition, half-/full-width folding, and squashing of long runs of a repeated character (such as a drawn-out laugh or a row of exclamation marks). It is meant to be called on raw text before [token_morph()] and friends; the tokenizers never apply it implicitly so that their behavior stays predictable.

Usage

```
text_normalize(
  x,
  squash = 2L,
  nfc = TRUE,
  width = c("halfwidth", "fullwidth", "none")
)
```

Arguments

x	A character vector. ‘NA‘ elements are returned unchanged.
squash	Maximum run length to keep for a repeated character. Runs longer than ‘squash‘ identical characters are collapsed to ‘squash‘ copies (a run of four identical characters becomes two at the default). Use a non-finite value such as ‘Inf‘ to disable squashing.
nfc	Apply Unicode NFC normalization so decomposed Hangul jamo are composed into single syllable characters.
width	Fold character width: ‘"halfwidth"‘ converts full-width Latin and digits to ASCII, ‘"fullwidth"‘ does the reverse, ‘"none"‘ leaves width untouched.

Value

A character vector the same length as ‘x‘.

Examples

```
text_normalize("\ubd84\uc11d \u314b\u314b\u314b\u314b \uc7ac\ubc0c\uc5b4\uc694!!!!")
text_normalize("\uff21\uff22\uff23\uff11\uff12\uff13", width = "halfwidth")
```

---

token_lemma *Tokenize Korean text into predicate dictionary forms*

Description

Analyzes text and returns the dictionary forms of its predicates (verbs and adjectives), following the [tokenizers][token_morph] contract. Fused tokens whose stem cannot be recovered are dropped; see [lemmatize_morphemes()] for the token-level detail and limitations.

Usage

```
token_lemma(
  phrase,
  keep = c("VV", "VA", "VX", "VCP", "VCN", "XSV", "XSA"),
  combine_root = TRUE,
  sys_dic = "",
  user_dic = "",
```

```
  parallel = FALSE,
  simplify = FALSE
)
```

Arguments

phrase	A character vector or list of character scalars.
keep	POS tags whose lemmas to keep. Compound tags match on their first component.
combine_root	Recombine a noun root with a following derivational suffix into one predicate lemma.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use parallel analysis for multiple documents.
simplify	When 'TRUE' and a single document is supplied, return a bare character vector instead of a length-one list.

Value

A list of character vectors of predicate lemmas, named when the input is named.

See Also

[lemmatize_morphemes()]

Examples

```
## Not run:
token_lemma(c("\uc544\uce68\uc744 \uba39\uc5c8\ub2e4",
              "\ub0a0\uc528\uac00 \uc88b\uc558\ub2e4"))

## End(Not run)
```

---

token_morph *Korean morpheme tokenizers*

Description

Tokenizes text using Korean POS categories. Filtering is applied to MeCab output rather than deleting punctuation or digits from the source text. The functions follow the **tokenizers** contract, so they drop directly into 'tidytext::unnest_tokens(token = token_nouns)' and related pipelines.

Usage

```
token_morph(
  phrase,
  strip_punct = FALSE,
  strip_numeric = FALSE,
  keep_pos = NULL,
  drop_pos = NULL,
  sys_dic = "",
  user_dic = "",
  parallel = FALSE,
  simplify = FALSE
)

token_words(
  phrase,
  strip_punct = FALSE,
  strip_numeric = FALSE,
  sys_dic = "",
  user_dic = "",
  parallel = FALSE,
  simplify = FALSE
)

token_nouns(
  phrase,
  strip_punct = FALSE,
  strip_numeric = FALSE,
  sys_dic = "",
  user_dic = "",
  parallel = FALSE,
  simplify = FALSE
)
```

Arguments

phrase	A character vector or list of character scalars.
strip_punct	Remove tokens tagged as Korean punctuation.
strip_numeric	Remove tokens tagged 'SN'.
keep_pos	Optional Korean POS tags to retain. Compound tags match when any component is selected.
drop_pos	Optional Korean POS tags to remove. Compound tags match when any component is selected. Combine with [stopwords_ko_tags()] to strip particles, endings, or other function morphemes.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use parallel analysis for multiple documents.

simplify	When 'TRUE' and a single document is supplied, return a bare character vector instead of a length-one list.

Value

A list of character vectors, one per document, named when the input is named. With 'simplify = TRUE' a single document returns a character vector.

`token_ngrams` *Korean morpheme n-grams and skip-grams*

Description

Creates n-grams after Korean morphological analysis. Stopwords split token sequences, so an n-gram never bridges across a removed stopword.

Usage

```
token_ngrams(
  phrase,
  n = 3L,
  div = c("morph", "words", "nouns"),
  stopwords = character(),
  ngram_delim = " ",
  skip = 0L,
  keep_pos = NULL,
  drop_pos = NULL,
  strip_punct = TRUE,
  strip_numeric = TRUE,
  sys_dic = "",
  user_dic = "",
  parallel = FALSE,
  simplify = FALSE
)
```

Arguments

phrase	A character vector or list of character scalars.
n	Positive integer n-gram sizes. Multiple values are supported.
div	Token preset: all morphemes, content words, or nouns.
stopwords	Character tokens that break n-gram sequences.
ngram_delim	Character scalar placed between terms.
skip	Non-negative exact numbers of tokens to skip between adjacent terms. 'skip = 0' creates contiguous n-grams.
keep_pos	Optional Korean POS tags to retain before n-gram generation.

drop_pos	Optional Korean POS tags to remove before n-gram generation.
strip_punct	Remove Korean punctuation tokens before generation.
strip_numeric	Remove 'SN' numeric tokens before generation.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use parallel morphological analysis for multiple documents.
simplify	When 'TRUE' and a single document is supplied, return a bare character vector instead of a length-one list.

Value

A list of character vectors, named when the input is named. Empty or too-short documents return 'character(0)'; missing documents return 'NA_character_'.

Examples

```
## Not run:
text <- "\ud55c\uad6d\uc5b4 \ud615\ud0dc\uc18c \ubd84\uc11d\uc744 \ud569\ub2c8\ub2e4"
token_ngrams(text, n = 2)
token_ngrams(text, n = 2:3, skip = 0:1)

## End(Not run)
```

---

words *Extract Korean content words*

Description

Keeps Korean POS categories beginning with 'N', 'V', 'M', or 'I', plus foreign-language tokens tagged 'SL'.

Usage

```
words(sentence, sys_dic = "", user_dic = "", parallel = FALSE)
```

Arguments

sentence	A character vector or list of character scalars.
sys_dic	Optional Korean system dictionary directory.
user_dic	Optional compiled user dictionary.
parallel	Use parallel analysis for multiple documents.

Value

A named list of character vectors.

Index

∗ **datasets**
demo_ko, 2
stopwords_ko, 11

demo_ko, 2
dict_add_words, 3
dict_compile *(*dict_add_words*)*, 3
dict_path *(*dict_add_words*)*, 3
dict_remove_words *(*dict_add_words*)*, 3
dict_use *(*dict_add_words*)*, 3
dict_words *(*dict_add_words*)*, 3

install_mecab, 4

keywords_textrank, 5
keywords_tfidf, 6
kwic, 7

lemmatize_morphemes, 8
lexicon_knu, 9

nouns, 10

pos, 11

stopwords_ko, 11
stopwords_ko_tags, 12
stopwords_ko_words, 13

text_normalize, 13
token_lemma, 14
token_morph, 15
token_ngrams, 17
token_nouns *(*token_morph*)*, 15
token_words *(*token_morph*)*, 15

words, 18