Package 'polyglotr'

June 8, 2026

Title Translate Text

Version 1.7.4

Description Provide easy methods to translate pieces of text.
Functions send requests to translation services online.

License MIT + file LICENSE

URL https://github.com/Tomeriko96/polyglotr/,
https://tomeriko96.github.io/polyglotr/

BugReports https://github.com/Tomeriko96/polyglotr/issues

Encoding UTF-8

RoxygenNote 7.3.3

Imports dplyr, httr, jsonlite, magrittr, purrr, RCurl, rlang, rvest,
stringr, tibble, urltools

Suggests knitr, rmarkdown, testthat (>= 3.0.0), text2vec, shiny,
shinydashboard, shinyjs, DT

Config/testthat/edition 3

VignetteBuilder knitr

Depends R (>= 3.5.0)

LazyData true

NeedsCompilation no

Author Tomer Iwan [aut, cre, cph]

Maintainer Tomer Iwan <iwantomer@hotmail.com>

Repository CRAN

Date/Publication 2026-06-08 08:20:02 UTC

Contents

`apertium_get_language_pairs`

Get Apertium Language Pairs

Description

This function retrieves the supported language pairs from the Apertium API.

Usage

```
apertium_get_language_pairs(host = "https://apertium.org/apy")
```

Arguments

`host`	Host URL for the Apertium API (default is "https://apertium.org/apy").

Value

A list of language pairs. Each element contains sourceLanguage and targetLanguage.

Examples

```
pairs <- apertium_get_language_pairs()
head(pairs, 5)
```

`apertium_translate` *Translate text using Apertium*

Description

Translate text using Apertium

Usage

```
apertium_translate(
  text,
  target_language,
  source_language,
  host = "https://apertium.org/apy"
)
```

Arguments

`text`	Text to translate. Can be a single string or a vector of strings.
`target_language`	Language to translate text to.
`source_language`	Language to translate text from.
`host`	Host URL for the Apertium API (default is "https://apertium.org/apy").

Value

Translated text. Returns a vector if input is a vector.

Examples

```
apertium_translate("Hello World", target_language = "es", source_language = "en")
apertium_translate("Hola mundo", target_language = "en", source_language = "es")

# Translate multiple texts
texts <- c("Hello", "Good morning", "Thank you")
apertium_translate(texts, target_language = "es", source_language = "en")
```

batch_translate *Batch Translation Function*

Description

This function translates a file into each target language using the polyglotr package's translate_file function, and saves the translated files.

Usage

```
batch_translate(input_file, source_language, target_languages)
```

Arguments

input_file	A character string indicating the path to the input file.
source_language	A character string indicating the source language.
target_languages	A character vector indicating the target languages.

Value

Nothing is returned.

Examples

```
## Not run:
batch_translate("README.md", "nl", c("fr", "es", "de"))

## End(Not run)
```

create_translation_table *Create a Translation Table*

Description

This function generates a translation table by translating a list of words into multiple languages.

Usage

```
create_translation_table(words, languages)
```

Arguments

words	A character vector containing the words to be translated.
languages	A character vector specifying the target languages for translation.

Value

A data frame representing the translation table with original words and translations in each language.

Examples

```
## Not run:
words <- c("Hello", "Translate", "Table", "Script")
languages <- c("es", "fr", "de", "nl")
translations <- create_translation_table(words, languages)
print(translations)

## End(Not run)
```

---

create_transliteration_table

Create a Transliteration Table

Description

This function generates a transliteration table by transliterating a list of words into multiple languages.

Usage

```
create_transliteration_table(words, languages)
```

Arguments

words	A character vector containing the words to be transliterated.
languages	A character vector specifying the target languages for transliteration.

Value

A data frame representing the transliteration table with original words and transliterations in each language.

Examples

```
## Not run:
words <- c("Hello world", "Goodbye", "Thank you", "Please")
languages <- c("ar", "he", "el", "ru", "fa")
transliterations <- create_transliteration_table(words, languages)
print(transliterations)

## End(Not run)
```

google_get_supported_languages

Get Supported Languages

Description

This function fetches the supported languages from the Google Cloud Translate documentation page.

Usage

```
google_get_supported_languages()
```

Value

A data frame containing the supported languages and their corresponding ISO 639-1 codes.

google_is_valid_language_code

Check if a language code is valid

Description

This function checks if a given language code is in the `google_supported_languages` dataset.

Usage

```
google_is_valid_language_code(language_code)
```

Arguments

`language_code`	The language code to check.

Value

A logical value indicating if the language code is valid.

Examples

```
## Not run:
google_is_valid_language_code("en") # TRUE
google_is_valid_language_code("fr") # TRUE
google_is_valid_language_code("xx") # FALSE

## End(Not run)
```

google_supported_languages *Google Supported Languages*

Description

This dataset contains the language names and iso codes of languages supported by Google Translate API.

Usage

```
google_supported_languages
```

Format

A data frame with two variables: language_name and iso_code

Source

Google Translate API

google_translate *Translate text using Google Translate*

Description

Translates input text to a specified language using the Google Translate mobile web interface. Automatically detects and preserves URLs by temporarily replacing them with placeholders. Long texts are split on word boundaries and translated in chunks, then reassembled.

Usage

```
google_translate(text, target_language = "en", source_language = "auto")
```

Arguments

text	This is the text that you want to translate. Can be a single string or a vector of strings.
target_language	This is the language that you want to translate the text into. The default value is "en" (English).
source_language	This is the language of the text to be translated. The default value is "auto", which attempts automatic language detection.

Value

A translated string or vector of translated strings, matching the length of the input.

Examples

```
# Translate a simple sentence
google_translate("I love languages", target_language = "es")

# Translate a vector of words
text_to_translate <- c("the", "quick", "brown")
google_translate(text_to_translate, "fr", "en")

# Translate text containing a URL
google_translate("Visit http://example.com for more info.", target_language = "de")
```

---

google_translate_long_text

Translate Long Text Using Google Translate

Description

Translates long text from one language to another using Google Translate by splitting the input into manageable chunks if necessary.

Usage

```
google_translate_long_text(
  text,
  target_language = "en",
  source_language = "auto",
  chunk_size = 1000,
  preserve_newlines = FALSE
)
```

Arguments

`text`	A single character string with the text to translate.
`target_language`	The language code to translate the text into (default: "en" for English).
`source_language`	The language code of the input text (default: "auto" for automatic detection).
`chunk_size`	Maximum number of characters per translation request (default: 1000).
`preserve_newlines`	Logical; if TRUE, preserves newlines between chunks in the output. If FALSE (default), newlines are replaced with spaces.

Value

A single character string containing the translated text.

Examples

```
## Not run:
long_text <- paste(rep("This is a long text to translate.", 100), collapse = " ")
google_translate_long_text(
  long_text,
  target_language = "de",
  source_language = "en",
  chunk_size = 500,
  preserve_newlines = TRUE
)

## End(Not run)
```

---

google_transliterate *Transliterate a single word or a sentence to the required language.*

Description

Transliterate a single word or a sentence to the required language.

Usage

```
google_transliterate(text, language_tag = "el", num = 5)
```

Arguments

text	The word or sentence to transliterate from Latin/Roman (English) script.
language_tag	The target language's ISO639 code. The default value for this argument is "el" for Greek.
num	The maximum number of suggestions to fetch. The default value for this argument is 5.

Value

Character vector of transliterated sentences or larger pieces of text.

Examples

```
## Not run:
google_transliterate("Hello world", "fr", 10)
google_transliterate("hello", "el", 10)

## End(Not run)
```

language_detect *Detect Language using Google Translate API*

Description

This function detects the language of a given text using the Google Translate API.

Usage

```
language_detect(text)
```

Arguments

text	The text for which the language needs to be detected.

Value

A character string representing the detected language.

launch_polyglotr_app *Launch polyglotr Shiny App*

Description

This function launches the Shiny web application for polyglotr, providing a user-friendly interface to access multiple translation services.

Usage

```
launch_polyglotr_app(launch.browser = TRUE, port = NULL, host = "127.0.0.1")
```

Arguments

launch.browser	Logical. Should the app be launched in the default browser? Default is TRUE.
port	The port number for the Shiny app. Default is NULL (auto-assign).
host	The host IP address. Default is "127.0.0.1".

Value

Starts the Shiny application. No return value.

Examples

```
## Not run:
# Launch the app in default browser
launch_polyglotr_app()

# Launch on specific port
launch_polyglotr_app(port = 3838)

# Launch without opening browser
launch_polyglotr_app(launch.browser = FALSE)

## End(Not run)
```

`microsoft_supported_languages` *Get the set of languages currently supported by the Microsoft Translator API*

Description

Get the set of languages currently supported by the Microsoft Translator API

Usage

```
microsoft_supported_languages(scope = NULL)
```

Arguments

`scope`	(optional) A comma-separated list of names defining the group of languages to return. Allowed group names are: translation, transliteration, and dictionary. If no scope is given, then all groups are returned.

Value

A list of supported languages for the specified groups.

Examples

```
## Not run:
microsoft_supported_languages(scope = "translation,transliteration,dictionary")

## End(Not run)
```

mymemory_translate *Translate text using mymemory translate*

Description

Translate text using mymemory translate

Usage

```
mymemory_translate(text, target_language = "en", source_language = "auto")
```

Arguments

text	Text to translate.
target_language	Language to translate text to.
source_language	Language to translate text from

Value

Translated text.

Examples

```
mymemory_translate("Hello World", target_language = "es", source_language = "en")
```

pons_dictionaries *Get the list of available dictionaries from PONS API*

Description

Get the list of available dictionaries from PONS API

Usage

```
pons_dictionaries(language = "en")
```

Arguments

language	The language of the output (ISO 639-1 - two-letter codes). Supported languages are de, el, en, es, fr, it, pl, pt, ru, sl, tr, zh.

Value

A list of available dictionaries in the specified language.

Examples

```
## Not run:
pons_dictionaries(language = "es")

## End(Not run)
```

`pons_translate` *Translate text using PONS*

Description

Translate text using PONS

Usage

```
pons_translate(text, target_language = "pt", source_language = "en")
```

Arguments

`text`	This is the text that you want to translate. Can be a single string or a vector of strings.
`target_language`	This is the language that you want to translate the text into. The default value for this argument is "pt" for Portuguese.
`source_language`	This is the language of the text that you want to translate. The default value for this argument is "en" for English.

Value

Translated text. If the input is a vector, it returns a character vector of translated strings.

Examples

```
## Not run:
pons_translate("I love languages!", target_language = "pt", source_language = "en")
text_to_translate <- c("The", "Greatest", "Language")
pons_translate(text_to_translate, "pt", "en")

## End(Not run)
```

qcri_api_key *Get the QCRI API key from the environment variable*

Description

Get the QCRI API key from the environment variable

Usage

```
qcri_api_key()
```

Value

The QCRI API key stored in the QCRI_API_KEY environment variable.

qcri_get_domains *QCRI Get Domains*

Description

This function retrieves the supported domains from the QCRI Multiterm API.

Usage

```
qcri_get_domains(api_key = qcri_api_key())
```

Arguments

api_key	The API key associated with the user account being used. If not provided, the function will attempt to retrieve it from the QCRI_API_KEY environment variable.

Value

A list with keys:

- success: Boolean indicating whether the request succeeded.
- domains: Array of supported domains, such as news, tedtalks etc. Only present if success is true.
- error: Error message in case success is false.

Examples

```
## Not run:
qcri_get_domains(api_key = "YourApiKey")
qcri_get_domains()

## End(Not run)
```

qcri_get_language_pairs *QCRI Get Language Pairs*

Description

This function retrieves the supported language pairs from the QCRI Multiterm API.

Usage

```
qcri_get_language_pairs(api_key = qcri_api_key())
```

Arguments

api_key	The API key associated with the user account being used. If not provided, the function will attempt to retrieve it from the QCRI_API_KEY environment variable. You can register for an API key at https://mt.qcri.org/api/register

Value

Language pairs.

Examples

```
## Not run:
qcri_get_language_pairs(api_key = "YourApiKey")
qcri_get_language_pairs()

## End(Not run)
```

qcri_translate_text *QCRI Translate Text*

Description

This function translates a text from the source language to the target language using the QCRI Multiterm API.

Usage

```
qcri_translate_text(text, langpair, domain, api_key = qcri_api_key())
```

Arguments

text	The text to be translated. This must be URL encoded.
langpair	The source-target language pair, where source is language of the provided text and target is the language into which the text has to be translated.
domain	The domain over which the translation is tuned.
api_key	The API key associated with the user account being used. If not provided, the function will attempt to retrieve it from the QCRI_API_KEY environment variable.

Value

Translated text.

Examples

```
## Not run:
qcri_translate_text(text = "Hello, world!",
langpair = "en-ar",
domain = "general",
api_key = "YourApiKey")
qcri_translate_text(text = "Hello, world!",
langpair = "en-ar",
domain = "general")

## End(Not run)
```

---

replace_urls_with_placeholders

Replace URLs in a sentence with placeholders

Description

Detects and replaces protocol-style links (e.g., `http://`, `https://`, `ftp://`) in a given sentence with unique placeholders like `__URL1__`, `__URL2__`, etc. This is useful for preparing text for translation or further processing while preserving original URLs.

Usage

```
replace_urls_with_placeholders(sentence)
```

Arguments

sentence	A character string potentially containing one or more URLs.

Value

A list with two elements:

text The input sentence with URLs replaced by placeholders.

urls A character vector of the extracted URLs.

`restore_urls_from_placeholders` *Restore URLs from placeholders in a translated text*

Description

Replaces placeholders like `__URL1__`, `__URL2__`, etc. in a translated sentence back with their original URLs. Handles both original and lowercased versions of the placeholder (to account for translation artifacts).

Usage

```
restore_urls_from_placeholders(translated, urls)
```

Arguments

`translated`	A character string where placeholders should be replaced with original URLs.
`urls`	A character vector of the original URLs to restore.

Value

A character string with the placeholders replaced by the corresponding URLs.

`translate_file` *Translate File*

Description

Translates the content of a file using Google Translate API.

Usage

```
translate_file(
  file_path,
  target_language = "en",
  source_language = "auto",
  overwrite = FALSE
)
```

Arguments

file_path	The path to the file to be translated.
target_language	The target language to translate the file content to. Default is "en".
source_language	The source language of the file content. Default is "auto".
overwrite	Logical indicating whether to overwrite the original file with the translated content. Default is FALSE.

Examples

```
## Not run:
translate_file("path/to/file.txt", target_language = "fr", source_language = "en", overwrite = TRUE)

## End(Not run)
```

---

translate_to_morse *Translate Text to Morse Code using the FunTranslations API*

Description

This function takes a string of text as input and translates it to Morse code using the FunTranslations API.

Usage

```
translate_to_morse(text, api_key = NULL)
```

Arguments

text	A character string containing the text to be translated to Morse code.
api_key	(optional) Your FunTranslations API key, if you have a paid subscription.

Value

A list containing the translated Morse code text and other metadata.

`translate_to_morse_audio` *Translate English Text to Morse Code with Audio*

Description

This function takes an English text string as input and translates it to Morse code with an audio output using the FunTranslations API.

Usage

```
translate_to_morse_audio(text, api_key = NULL)
```

Arguments

`text`	A character string containing the English text to be translated.
`api_key`	(optional) Your FunTranslations API key, if you have a paid subscription.

Value

A list containing the translated Morse code text, the Morse code audio as a base64-encoded string, and other metadata.

`wikimedia_detect_language` *Detect the language of a text*

Description

This function sends a POST request to the Wikimedia Language ID API with the specified text, parses the JSON response, and returns the detected language.

Usage

```
wikimedia_detect_language(text)
```

Arguments

`text`	The text whose language is to be detected.

Value

The detected language.

Examples

```
{

# Detect the language of a text
wikimedia_detect_language("Hallo, wereld")

}
```

`wikipedia_get_language_names` *Get language names*

Description

This function sends a GET request to the Wikipedia API and returns the language names as a dataframe.

Usage

```
wikipedia_get_language_names()
```

Value

A dataframe of language names.

Examples

```
wikipedia_get_language_names()
```

`wmcloud_translate` *Translate content using WMCloud*

Description

This function sends a POST request to the WMCloud translation API with the specified parameters, parses the JSON response, and returns the translated content.

Usage

```
wmcloud_translate(
  content,
  target_language = "en",
  source_language = "en",
  format = "text",
  model = "nllb200-600M"
)
```

Arguments

content	The content to translate. Can be plain text, a URL (for a webpage), a JSON string, or a Markdown string.
target_language	The target language for the translation (default is "en").
source_language	The source language of the content (default is "en").
format	The format of the content ("json", "markdown", "text", "webpage").
model	The model to use for the translation (only "nllb200-600M" is currently known to work).

Value

The translated content.

Examples

```
## Not run:
# Translate plain text
wmcloud_translate("rijst",
target_language = "es",
source_language = "nl", format = "text")

# Translate a webpage
wmcloud_translate("https://en.m.wikivoyage.org/wiki/Goes",
target_language = "es",
source_language = "en", format = "webpage")

# Translate JSON content
wmcloud_translate('{
    "id": 1,
    "title": "Chicken Biryani",
    "description": "Chicken Biryani is a savory chicken and rice dish",
    "ingredients": [ "Vegetable oil", "Garlic", "Ginger" ,"Rice"]
}
               ', target_language = "es", source_language = "en", format = "json")

# Translate Markdown content
wmcloud_translate('# Heading

This is a [link to Wikipedia](https://wikipedia.org)
                ', target_language = "es", source_language = "en", format = "markdown")

## End(Not run)
```

Index