Package 'refinr'

April 24, 2022

Title Cluster and Merge Similar Values Within a Character Vector

Version 0.3.2

Description These functions take a character vector as input, identify and cluster similar values, and then merge clusters together so their values become identical. The functions are an implementation of the key collision and ngram fingerprint algorithms from the open source tool Open Refine <https://openrefine.org/>. More info on key collision and ngram fingerprint can be found here <https://docs.openrefine.org/next/technical-reference/clustering-in-depth/>.

Depends R (>= 3.0.2)

License GPL-3

Encoding UTF-8

Imports Rcpp, stringdist (>= 0.9.5.1), stringi

RoxygenNote 7.1.1

LinkingTo Rcpp, stringdist (>= 0.9.5.1)

URL https://github.com/ChrisMuir/refinr

BugReports https://github.com/ChrisMuir/refinr/issues

Suggests testthat, knitr, rmarkdown, dplyr

VignetteBuilder knitr

NeedsCompilation yes

Author Chris Muir [aut, cre]

Maintainer Chris Muir <chrismuirRVA@gmail.com>

Repository CRAN

Date/Publication 2022-04-24 02:20:05 UTC

R topics documented:

`key_collision_merge` *Value merging based on Key Collision*

Description

This function takes a character vector and makes edits and merges values that are approximately equivalent yet not identical. It clusters values based on the key collision method, described here https://docs.openrefine.org/next/technical-reference/clustering-in-depth/.

Usage

```
key_collision_merge(
  vect,
  ignore_strings = NULL,
  bus_suffix = TRUE,
  dict = NULL
)
```

Arguments

`vect`	Character vector, items to be potentially clustered and merged.
`ignore_strings`	Character vector, these strings will be ignored during the merging of values within `vect`. Default value is NULL.
`bus_suffix`	Logical, indicating whether the merging of records should be insensitive to common business suffixes or not. Default value is TRUE.
`dict`	Character vector, meant to act as a dictionary during the merging process. If any items within `vect` have a match in dict, then those items will always be edited to be identical to their match in dict. Default value is NULL.

Value

Character vector with similar values merged.

Examples

```
x <- c("Acme Pizza, Inc.", "ACME PIZZA COMPANY", "pizza, acme llc",
       "Acme Pizza, Inc.")
key_collision_merge(vect = x)

# Use parameter "dict" to influence how clustered values are edited.
key_collision_merge(vect = x, dict = c("Nicks Pizza", "acme PIZZA inc"))

# Use parameter 'ignore_strings' to ignore specific strings during merging
# of values.
x <- c("Bakersfield Highschool", "BAKERSFIELD high",
       "high school, bakersfield")
key_collision_merge(x, ignore_strings = c("high", "school", "highschool"))
```

n_gram_merge *Value merging based on ngram fingerprints*

Description

This function takes a character vector and makes edits and merges values that are approximately equivalent yet not identical. It uses a two step process, the first is clustering values based on their ngram fingerprint (described here https://docs.openrefine.org/next/technical-reference/clustering-in-depth/). The second step is merging values based on approximate string matching of the ngram fingerprints, using the [sd_lower_tri()] C function from the package `stringdist`.

Usage

```
n_gram_merge(
  vect,
  numgram = 2,
  ignore_strings = NULL,
  bus_suffix = TRUE,
  edit_threshold = 1,
  weight = c(d = 0.33, i = 0.33, s = 1, t = 0.5),
  ...
)
```

Arguments

`vect`	Character vector, items to be potentially clustered and merged.
`numgram`	Numeric value, indicating the number of characters that will occupy each ngram token. Default value is 2.
`ignore_strings`	Character vector, these strings will be ignored during the merging of values within `vect`. Default value is NULL.
`bus_suffix`	Logical, indicating whether the merging of records should be insensitive to common business suffixes or not. Default value is TRUE.
`edit_threshold`	Numeric value, indicating the threshold at which a merge is performed, based on the sum of the edit values derived from param `weight`. Default value is 1. If this parameter is set to 0 or NA, then no approximate string matching will be done, and all merging will be based on strings that have identical ngram fingerprints.
`weight`	Numeric vector, indicating the weights to assign to the four edit operations (see details below), for the purpose of approximate string matching. Default values are c(d = 0.33, i = 0.33, s = 1, t = 0.5). This parameter gets passed along to the `stringdist` function. Must be either a numeric vector of length four, or NA.
`...`	additional args to be passed along to the `stringdist` function. The acceptable args are identical to those of [stringdistmatrix()].

Details

The values of arg `weight` are edit distance values that get passed to the `stringdist` edit distance function. The param takes four arguments, each one is a specific type of edit, with default penalty value.

- d: deletion, default value is 0.33
- i: insertion, default value is 0.33
- s: substitution, default value is 1
- t: transposition, default value is 0.5

Value

Character vector with similar values merged.

Examples

```
x <- c("Acme Pizza, Inc.", "ACME PIZA COMPANY", "Acme Pizzazza LLC")

n_gram_merge(vect = x)

# The performance of the approximate string matching can be ajusted using
# parameters 'weight' or 'edit_threshold'
n_gram_merge(vect = x,
             weight = c(d = 0.4, i = 1, s = 1, t = 1))

# Use parameter 'ignore_strings' to ignore specific strings during merging
# of values.
x <- c("Bakersfield Highschool", "BAKERSFIELD high",
       "high school, bakersfield")
n_gram_merge(vect = x, ignore_strings = c("high", "school", "highschool"))
```

---

refinr *Cluster and Merge Similar Values Within a Character Vector*

Description

These functions take a character vector as input, identify and cluster similar values, and then merge clusters together so their values become identical. The functions are an implementation of the key collision and ngram fingerprint algorithms from the open source tool Open Refine.

Documentation for Open Refine

- Open Refine Site https://openrefine.org/
- Details on Open Refine clustering algorithms https://docs.openrefine.org/next/technical-reference/clustering-in-depth/

Development links

- https://github.com/ChrisMuir/refinr
- Report bugs at https://github.com/ChrisMuir/refinr/issues

refinr features the following functions

- key_collision_merge
- n_gram_merge

Index