

Package ‘firmmatchr’

March 6, 2026

Title Robust Probabilistic Matching for German Company Names

Version 0.1.3

Description

A pipeline for matching messy company name strings against a clean dictionary (e.g., 'Orbis'). Implements a cascading strategy: Exact -> Fuzzy ('zoomerjoin') -> 'FTS5' ('SQLite') -> Rarity Weighted. References: Beniamino Green (2025) <<https://beniamino.org/zoomerjoin/>>; <<https://www.sqlite.org/fts5.html>>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports data.table, stringi, stringdist, zoomerjoin, DBI, RSQLite, cli, progressr, httr, jsonlite, glue, purrr, readr, dplyr

Suggests testthat

NeedsCompilation no

Author Giulian Etingin-Frati [aut, cre]

Maintainer Giulian Etingin-Frati <etingin-frati@kof.ethz.ch>

Repository CRAN

Date/Publication 2026-03-06 14:20:02 UTC

Contents

match_companies	2
normalize_company_name	3
validate_matches_llm	4
Index	6

match_companies	<i>Match Company Names against a Dictionary</i>
-----------------	---

Description

Runs a cascading matching pipeline: Exact -> Fuzzy (Zoomer) -> FTS5 -> Rarity. Matches found in earlier steps are removed from subsequent steps.

Usage

```
match_companies(  
  queries,  
  dictionary,  
  query_col = "company_name",  
  dict_col = "company_name",  
  unique_id_col = "query_id",  
  dict_id_col = "orbis_id",  
  threshold_jw = 0.8,  
  threshold_zoomer = 0.4,  
  threshold_rarity = 1,  
  n_cores = 1  
)
```

Arguments

queries	Data frame. Must contain columns specified in query_col and unique_id_col.
dictionary	Data frame. Must contain columns specified in dict_col and dict_id_col.
query_col	String. Column name for company names in queries.
dict_col	String. Column name for company names in dictionary.
unique_id_col	String. ID column in queries.
dict_id_col	String. ID column in dictionary.
threshold_jw	Numeric (0-1). Minimum Jaro-Winkler similarity. Default 0.8.
threshold_zoomer	Numeric (0-1). Jaccard threshold for blocking. Default 0.4.
threshold_rarity	Numeric. Minimum score for rarity matching. Default 1.0.
n_cores	Integer. Number of cores (reserved for future parallel implementation).

Value

A data.table containing query_id, dict_id, and match_type.

Examples

```
# Create sample query data
queries <- data.frame(
  query_id = 1:3,
  company_name = c("BMW", "Siemens AG", "Deutsche Bank")
)

# Create sample dictionary
dictionary <- data.frame(
  orbis_id = c("D001", "D002", "D003"),
  company_name = c("BMW AG", "Siemens Aktiengesellschaft", "Commerzbank AG")
)

# Match companies (uses multi-threaded Rust internals via zoomerjoin)

results <- match_companies(
  queries = queries,
  dictionary = dictionary,
  query_col = "company_name",
  dict_col = "company_name",
  unique_id_col = "query_id",
  dict_id_col = "orbis_id"
)

print(results)
```

normalize_company_name

Normalize Company Names

Description

Standardizes company names by lowercasing, removing legal suffixes, translating characters to ASCII, and removing noise words.

Usage

```
normalize_company_name(x)
```

Arguments

x A character vector of company names.

Value

A character vector of normalized names.

Examples

```
# Normalize a single company name
normalize_company_name("BMW AG")
normalize_company_name("Siemens GmbH & Co. KG")

# Normalize multiple names
companies <- c("Deutsche Bank AG", "VW Group", "BASF SE")
normalize_company_name(companies)
```

validate_matches_llm *Validate Matches using LLM (Azure OpenAI)*

Description

Sends doubtful matches (not "Perfect" or "Unmatched") to an LLM for verification. Supports resuming from interruptions via chunk files.

Usage

```
validate_matches_llm(
  data,
  query_name_col,
  dict_name_col,
  output_dir = tempdir(),
  filename_stem = "match_validation",
  batch_size = 20,
  api_key = NULL,
  endpoint = NULL,
  deployment = NULL,
  engine = c("azure", "openai", "local")
)
```

Arguments

data	Data frame. Must contain the columns specified by query_name_col and dict_name_col.
query_name_col	String. Column containing the user's query name (Employer).
dict_name_col	String. Column containing the dictionary match name (Registry).
output_dir	String. Directory to save temporary chunks and final results. Defaults to tempdir().
filename_stem	String. Base name for output files.
batch_size	Integer. Number of rows to process before saving a chunk.
api_key	String. API Key. Defaults to Sys.getenv("AZURE_API_KEY") or Sys.getenv("OPENAI_API_KEY").
endpoint	String. API Endpoint. Defaults to Sys.getenv("AZURE_ENDPOINT") or Sys.getenv("OPENAI_ENDPOINT").
deployment	String. Deployment or model name. Defaults to Sys.getenv("AZURE_DEPLOYMENT") or Sys.getenv("OPENAI_MODEL").
engine	String. Either "azure", "openai", or "local". Defaults to "azure". Use "local" (or "openai") for local LLMs like Ollama.

Value

A data frame with added LLM_decision and LLM_reason columns.

Examples

```
## Not run:
# Sample matched data
matched_data <- data.frame(
  employer_name = c("BMW", "Siemens"),
  registry_name = c("BMW AG", "SAP SE"),
  dict_id = c("D001", "D002"),
  match_type = c("Fuzzy", "Fuzzy")
)

# Validate using LLM (requires Azure credentials)
validated <- validate_matches_llm(
  data = matched_data,
  query_name_col = "employer_name",
  dict_name_col = "registry_name"
)

print(validated)

## End(Not run)
```

Index

`match_companies`, [2](#)

`normalize_company_name`, [3](#)

`validate_matches_llm`, [4](#)