

Package ‘ragR’

October 25, 2026

Title Retrieval-Augmented Generation and RAG Evaluation Tools in R

Version 0.1.1

Description Provides tools for document ingestion, embedding storage, retrieval-augmented generation (RAG), and RAG evaluation for question answering systems. The package includes an R-native vector store, wrappers for OpenAI embedding and chat-completion APIs, QA logging utilities, and LLM-scored RAGAS-style evaluation metrics.

License GPL-3

URL <https://github.com/aimalrehman92/ragR>

BugReports <https://github.com/aimalrehman92/ragR/issues>

Encoding UTF-8

Language en-US

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports dplyr, httr2, jsonlite, pdftools, readtext, tibble

Suggests plumber, stringr, yaml, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Muhammad Aimal Rehman [aut, cre],
Zhili Lu [aut],
Seoyoung Kim [aut],
Chi-Kuang Yeh [aut]

Maintainer Muhammad Aimal Rehman <rehman.aimal@gmail.com>

Repository CRAN

Date/Publication 2026-10-25 10:00:02 UTC

Contents

ragR-package	2
api_chat_handler	5
api_clear_all_handler	5
api_clear_handler	6
api_ragas_clear_handler	6
api_ragas_handler	7
api_ragas_report_handler	7
chunk_text_sentence	8
clear_qa_log	8
clear_qa_metrics	9
compute_ragas_metrics	9
compute_ragas_metrics_llm	10
generate_openai_chat	11
generate_ragas_report	12
generate_ragas_report_llm	12
get_openai_embeddings	13
ingest_documents	14
load_qa_log	15
load_qa_metrics	16
load_rag_config	16
log_rag_interaction	17
plot_ragas_means	18
qa_log_empty	18
qa_metrics_empty	19
query_rag	19
save_qa_log	20
save_qa_metrics	21
summarize_ragas	21
vectorstore_clear_all	22
vectorstore_delete_collection	22
vectorstore_load	23
vectorstore_query	23
vectorstore_upsert	24
Index	25

 ragR-package

ragR: Retrieval-Augmented Generation + RAGAS Evaluation in R

Description

The **ragR** package provides a Retrieval-Augmented Generation (RAG) system implemented in R. It includes:

- A data ingestion pipeline for PDF, text, and Word documents
- An R-native vector store for chunk embeddings

- A configurable RAG pipeline that uses OpenAI embeddings and chat models
- Persistent QA logging for reproducible evaluation
- LLM-scored RAGAS-style metrics, including context precision, context recall, answer relevance, and faithfulness
- A Plumber-based HTTP API suitable for chatbot and evaluation frontends

Architecture

The package is organized into several modules:

- **Ingestion** (`ingestion_files.R`) Extracts text from PDF/TXT/DOCX files, chunks it, and prepares data for embedding and storage. Ingestion is handled through package functions and development scripts, not through the minimal HTTP API.
- **Embeddings & Chat** (`embeddings_openai.R`) Wraps the OpenAI REST API for embeddings, such as "text-embedding-3-small", and chat models, such as "gpt-4o-mini". Reads the API key from `OPENAI_API_KEY`.
- **Vector Store** (`vectorstore_interface.R`) Implements an R-native vector store using tibbles and numeric matrices, with functions to add, query, and clear stored embeddings. Also includes `vectorstore_clear_all()` for wiping the entire vector store in one operation.
- **RAG Pipeline** (`rag_pipeline.R`) Given a user question, the pipeline:
 1. Computes an embedding for the question
 2. Retrieves top-k similar chunks from the vector store
 3. Builds a grounded context-aware prompt
 4. Calls the OpenAI chat model to generate an answer
- **QA Logging & Persistence** (`qa_logging.R`, `qa_persistence.R`) Stores each interaction in `db/qa_log.rds`, including the question, model answer, retrieved chunks, final grounded prompt, model names, and timestamps. Provides helpers to load, save, and clear logs.
- **RAGAS Metrics & Summary** (`ragas_metrics.R`, `ragas_summary.R`) Computes LLM-scored RAGAS-style metrics, including context precision, context recall, answer relevance, faithfulness, and an overall score. Provides summary helpers for aggregating metric results across QA pairs.
- **Reports** (`ragas_report.R`) Generates a performance report by:
 - Computing LLM-scored metrics from the QA log
 - Saving `db/qa_metrics.rds`
 - Writing `reports/ragas/ragas_summary.csv`
 - Writing a bar chart of mean metrics to `reports/ragas/ragas_means.png`
- **API Handlers** (`api_handlers.R`) Functions that connect the core logic to HTTP endpoints. These are called by the Plumber router defined in `inst/api/`.

HTTP API

When `scripts/dev/run_api.R` is executed, a Plumber server exposes:

- `POST /chat` Runs the RAG pipeline for a user question and logs the result into `db/qa_log.rds`.

- GET /ragas Computes and returns LLM-scored RAGAS-style metrics and a summary for the logged QA pairs.
- POST /ragas/report Computes LLM-scored metrics from the QA log, saves the metrics and report artifacts to disk, and returns a small summary object.
- POST /ragas/clear Clears the QA log, QA metrics, and report files.
- POST /clear Clears one vector-store collection.
- POST /clear_all Clears the entire vector store across all collections. Uses `api_clear_all_handler()`, which calls `vectorstore_clear_all()`.

Usage

Programmatic usage inside R:

```
library(ragR)

# Run a RAG query
res <- query_rag(
  question      = "What is this document about?",
  collection     = "default",
  top_k         = 5,
  embedding_model = "text-embedding-3-small",
  chat_model     = "gpt-4o-mini",
  system_prompt  = "You are a helpful assistant."
)

cat(res$answer)
```

As a backend service:

```
# From the project root
source("scripts/dev/run_api.R")
# Visit http://127.0.0.1:8000/__docs__/ for interactive docs
```

Author(s)

Maintainer: Muhammad Aimal Rehman <rehman.aimal@gmail.com>

Authors:

- Zhili Lu
- Chi-Kuang Yeh

See Also

Useful links:

- <https://github.com/aimalrehman92/ragR>
- Report bugs at <https://github.com/aimalrehman92/ragR/issues>

api_chat_handler	<i>API handler: chat with the RAG pipeline</i>
------------------	--

Description

This handler receives a parsed JSON request body, runs the RAG pipeline, appends the interaction to the QA log, and returns the model answer.

Usage

```
api_chat_handler(body)
```

Arguments

body	Parsed JSON request body.
------	---------------------------

Value

A list suitable for JSON serialization.

api_clear_all_handler	<i>API handler: clear the entire vector store</i>
-----------------------	---

Description

This handler wipes the R-native vector store by calling `vectorstore_clear_all()`, removing all collections and stored embeddings.

Usage

```
api_clear_all_handler()
```

Value

A list with elements status and message, suitable for JSON serialization.

api_clear_handler	<i>API handler: clear one vector-store collection</i>
-------------------	---

Description

This handler deletes one collection from the R-native vector store. If no collection is supplied, "default" is used.

Usage

```
api_clear_handler(body = NULL)
```

Arguments

body	Parsed JSON request body, or NULL.
------	------------------------------------

Value

A list with status, collection, and message.

api_ragas_clear_handler	<i>API handler: clear RAGAS/QA evaluation files</i>
-------------------------	---

Description

Clears the saved QA log, QA metrics file, and generated RAGAS report files.

Usage

```
api_ragas_clear_handler(
  qa_log_path = "db/qa_log.rds",
  qa_metrics_path = "db/qa_metrics.rds",
  output_dir = "reports/ragas"
)
```

Arguments

qa_log_path	Character scalar; path to the QA log RDS file.
qa_metrics_path	Character scalar; path to the QA metrics RDS file.
output_dir	Character scalar; directory containing RAGAS report files.

Value

A list with status and message.

api_ragas_handler	<i>API handler: compute RAGAS metrics and summary from saved QA log</i>
-------------------	---

Description

API handler: compute RAGAS metrics and summary from saved QA log

Usage

```
api_ragas_handler(path = "db/qa_log.rds", judge_model = "gpt-4o-mini")
```

Arguments

- path Character scalar; path to the QA log RDS file. Defaults to "db/qa_log.rds".
- judge_model Character scalar; judge model used for LLM-scored metrics. Defaults to "gpt-4o-mini".

Value

A list with status, n_qa, metrics, summary.

api_ragas_report_handler	<i>API handler: generate RAGAS performance report</i>
--------------------------	---

Description

API handler: generate RAGAS performance report

Usage

```
api_ragas_report_handler(  
  qa_log_path = "db/qa_log.rds",  
  qa_metrics_path = "db/qa_metrics.rds",  
  output_dir = "reports/ragas",  
  judge_model = "gpt-4o-mini"  
)
```

Arguments

- qa_log_path Character scalar; path to the QA log RDS file. Defaults to "db/qa_log.rds".
- qa_metrics_path Character scalar; path to the QA metrics RDS file. Defaults to "db/qa_metrics.rds".
- output_dir Character scalar; directory where outputs are written. Defaults to "reports/ragas".
- judge_model Character scalar; judge model used for LLM-scored metrics. Defaults to "gpt-4o-mini".

Value

A list with status, n_qa, qa_metrics_path, summary_csv_path, and plot_path.

chunk_text_sentence	<i>Chunk text into sentences (strict)</i>
---------------------	---

Description

Splits text into individual sentences. Each returned element is intended to be exactly one sentence (best-effort based on punctuation).

Usage

```
chunk_text_sentence(text)
```

Arguments

text	Character scalar.
------	-------------------

Value

Character vector; one sentence per element.

clear_qa_log	<i>Clear the QA log on disk</i>
--------------	---------------------------------

Description

This helper overwrites the QA log file with an empty QA log, as created by [qa_log_empty\(\)](#). It is intended for starting a fresh evaluation session.

Usage

```
clear_qa_log(path = "db/qa_log.rds")
```

Arguments

path	Character scalar; path to the QA log RDS file. Defaults to "db/qa_log.rds".
------	---

Value

Invisibly, the path to the saved file.

clear_qa_metrics	<i>Clear QA metrics on disk</i>
------------------	---------------------------------

Description

This helper overwrites the QA metrics file with an empty metrics tibble, as created by `qa_metrics_empty()`.

Usage

```
clear_qa_metrics(path = "db/qa_metrics.rds")
```

Arguments

path Character scalar; path to the QA metrics RDS file. Defaults to "db/qa_metrics.rds".

Value

Invisibly, the path to the saved file.

compute_ragas_metrics	<i>Compute RAGAS metrics</i>
-----------------------	------------------------------

Description

Convenience wrapper for computing LLM-scored RAGAS-style metrics.

Usage

```
compute_ragas_metrics(  
  qa_log,  
  judge_model = "gpt-4o-mini",  
  seed = NULL,  
  embedding_model = "text-embedding-3-small",  
  answer_relevance_strictness = 3L  
)
```

Arguments

qa_log QA log tibble.
judge_model Judge model for LLM-scored metrics.
seed Optional integer forwarded to LLM-scored metrics.
embedding_model Embedding model for answer relevance.
answer_relevance_strictness Number of reverse questions generated for answer relevance.

Value

A metrics tibble.

compute_ragas_metrics_llm
<i>Compute RAGAS-style metrics (LLM scored)</i>

Description

This implementation mirrors the structured Python RAGAS workflow rather than asking the judge model for one direct scalar score per metric.

Usage

```
compute_ragas_metrics_llm(  
  qa_log,  
  judge_model = "gpt-4o-mini",  
  seed = NULL,  
  embedding_model = "text-embedding-3-small",  
  answer_relevance_strictness = 3L  
)
```

Arguments

qa_log	A tibble created and populated by <code>log_rag_interaction()</code> .
judge_model	Character scalar; judge model name.
seed	Optional integer forwarded to judge model calls.
embedding_model	Embedding model name used for answer relevance.
answer_relevance_strictness	Number of generated reverse questions for answer relevance. Python RAGAS defaults to 3.

Details

- Notes:
- context_precision uses answer_reference if available; otherwise it uses answer_model, matching the Python with-reference / without-reference split.
 - answer_relevance requires an embedding helper to be available in the package environment.

Value

A tibble with one row per qa_id and metric columns.

generate_openai_chat	<i>Generate a chat completion from OpenAI</i>
----------------------	---

Description

Used by the RAG pipeline and RAGAS-style metric prompts to generate text from a constructed prompt.

Usage

```
generate_openai_chat(  
    prompt,  
    model = "gpt-4o-mini",  
    system_message = NULL,  
    temperature = 0,  
    max_output_tokens = 512L,  
    seed = NULL,  
    api_key = NULL  
)
```

Arguments

prompt	Character scalar; the user/content prompt.
model	Character scalar; chat model name, e.g., "gpt-4o-mini".
system_message	Optional API-level system message. If NULL or empty, no system message is sent.
temperature	Numeric; sampling temperature.
max_output_tokens	Integer or NULL; maximum tokens to generate. If NULL, the parameter is omitted from the request.
seed	Optional integer. If provided and supported by the model/provider, it is sent as seed to encourage reproducible outputs.
api_key	OpenAI API key. If NULL, uses the OPENAI_API_KEY environment variable.

Value

Character scalar containing the model's answer.

`generate_ragas_report` *Generate a RAGAS performance report*

Description

Loads a QA log from disk, computes LLM-scored RAGAS-style metrics, and writes:

- metrics RDS (`qa_metrics_path`)
- summary CSV (`ragas_summary.csv`)
- means bar plot PNG (`ragas_means.png`)

Usage

```
generate_ragas_report(  
  qa_log_path = "db/qa_log.rds",  
  qa_metrics_path = "db/qa_metrics.rds",  
  output_dir = "reports/ragas",  
  judge_model = "gpt-4o-mini"  
)
```

Arguments

<code>qa_log_path</code>	Path to QA log RDS (default: "db/qa_log.rds").
<code>qa_metrics_path</code>	Path to metrics RDS output (default: "db/qa_metrics.rds").
<code>output_dir</code>	Output directory for CSV/PNG (default: "reports/ragas").
<code>judge_model</code>	Judge model used for LLM scoring (default: "gpt-4o-mini").

Value

A list with: `status`, `n_qa`, `qa_metrics_path`, `summary_csv_path`, `plot_path`.

`generate_ragas_report_llm`
Generate RAGAS report using LLM scoring

Description

This is an explicit alias for `generate_ragas_report()`.

Usage

```
generate_ragas_report_llm(
  qa_log_path = "db/qa_log.rds",
  qa_metrics_path = "db/qa_metrics.rds",
  output_dir = "reports/ragas",
  judge_model = "gpt-4o-mini"
)
```

Arguments

qa_log_path	Path to QA log RDS (default: "db/qa_log.rds").
qa_metrics_path	Path to metrics RDS output (default: "db/qa_metrics.rds").
output_dir	Output directory for CSV/PNG (default: "reports/ragas").
judge_model	Judge model used for LLM scoring (default: "gpt-4o-mini").

Value

A list with: status, n_qa, qa_metrics_path, summary_csv_path, plot_path.

get_openai_embeddings *Get embeddings from OpenAI*

Description

Calls the OpenAI embeddings endpoint to obtain vector representations for one or more input texts. Used by both ingestion and query-time retrieval.

Usage

```
get_openai_embeddings(texts, model = "text-embedding-3-small", api_key = NULL)
```

Arguments

texts	Character vector of texts to embed.
model	Character scalar; embedding model name, e.g., "text-embedding-3-small".
api_key	OpenAI API key. If NULL, uses the OPENAI_API_KEY environment variable.

Value

A numeric matrix with one row per input text.

ingest_documents	<i>Ingest documents into the vector store</i>
------------------	---

Description

Reads PDF, DOCX, or TXT files, extracts text, cleans it lightly, chunks it, embeds chunks, and stores them in a vector store collection.

Usage

```
ingest_documents(
    paths,
    collection = "default",
    chunk_size = 500L,
    chunk_overlap = 50L,
    chunking_strategy = c("character", "sentence"),
    embedding_model = "text-embedding-3-small",
    embedding_batch_size = 128L,
    embedding_max_chars = 8000L,
    retry = TRUE,
    max_retries = 5L,
    resume = TRUE,
    checkpoint_path = NULL,
    use_openai = TRUE,
    verbose = TRUE
)
```

Arguments

paths	Character vector of file paths to ingest.
collection	Character scalar; name of the collection.
chunk_size	Integer; target chunk size (in characters). Used for "character".
chunk_overlap	Integer; overlap between consecutive chunks. Used for "character".
chunking_strategy	Character; "character" or "sentence".
embedding_model	Character; OpenAI embedding model name.
embedding_batch_size	Integer; number of chunks per embeddings request. Default 128. Internally capped for safety.
embedding_max_chars	Integer; hard cap (in characters) applied to each chunk before embedding to avoid request failures. Default 8000.
retry	Logical; whether to retry transient API failures. Default TRUE.
max_retries	Integer; maximum retries per batch (transient failures). Default 5.

resume	Logical; whether to use a local checkpoint to skip chunks already embedded in a previous run. Default TRUE.
checkpoint_path	Optional character scalar; file path for the resume checkpoint. If NULL, uses a temp file based on the collection name.
use_openai	Logical; if TRUE, use OpenAI embeddings. If FALSE, use a local dummy embedding generator (for offline development / tests).
verbose	Logical; if TRUE, log progress messages.

Details

Chunking strategies:

- "character": overlapping fixed-size character windows.
- "sentence": strict sentence splitting via `chunk_text_sentence()`.

Cleaning strategy (minimal, deterministic):

- Replace carriage returns and newlines with spaces
- Remove control characters
- Collapse multiple whitespace to a single space
- Trim leading/trailing whitespace

Embedding strategy (robust by default):

- Embeddings are computed in batches (default `embedding_batch_size` = 128)
- Batches are retried with exponential backoff on transient failures (e.g., 429/5xx)
- If a 400 occurs due to request size, the batch is automatically split
- Each chunk is truncated to a safe maximum length before embedding
- Optional resumable ingestion via a local checkpoint file

Value

A tibble with a per-file ingestion summary (`collection`, `path`, `n_chunks`).

load_qa_log	<i>Load a QA log from disk</i>
-------------	--------------------------------

Description

This helper loads a QA log tibble from an RDS file. If the file does not exist, it returns an empty QA log created by `qa_log_empty()`.

Usage

```
load_qa_log(path = "db/qa_log.rds")
```

Arguments

path Character scalar; path to the RDS file. Defaults to "db/qa_log.rds".

Value

A tibble containing the QA log.

load_qa_metrics	<i>Load QA metrics from disk</i>
-----------------	----------------------------------

Description

This helper loads QA metrics from an RDS file. If the file does not exist, it returns an empty metrics tibble created by `qa_metrics_empty()`.

Usage

```
load_qa_metrics(path = "db/qa_metrics.rds")
```

Arguments

path Character scalar; path to the RDS file. Defaults to "db/qa_metrics.rds".

Value

A tibble containing the QA metrics.

load_rag_config	<i>Load RAG configuration</i>
-----------------	-------------------------------

Description

Placeholder for a configuration loader. In the future, this can read a YAML file and merge it with environment variables and defaults.

Usage

```
load_rag_config(path = NULL)
```

Arguments

path Optional path to a YAML config file. Currently unused.

Value

A list of configuration values. Currently returns an empty list.

log_rag_interaction	<i>Append one RAG interaction to the QA log</i>
---------------------	---

Description

Adds one row to an in-memory QA log tibble.

Usage

```
log_rag_interaction(  
  qa_log,  
  question,  
  rag_result,  
  collection,  
  chat_model = "gpt-4o-mini",  
  embedding_model = "text-embedding-3-small",  
  qa_id = NULL,  
  timestamp = Sys.time()  
)
```

Arguments

qa_log	Existing QA log tibble, or NULL.
question	Character scalar.
rag_result	List returned by query_rag() . Must include answer; may include retrieved and prompt.
collection	Collection name used for retrieval.
chat_model	Chat model name.
embedding_model	Embedding model name.
qa_id	Optional integer QA id. If NULL, auto-increments.
timestamp	POSIXct timestamp.

Value

Updated QA log tibble.

plot_ragas_means	<i>Plot mean RAGAS metrics to a PNG</i>
------------------	---

Description

Creates a simple bar chart of mean metric values.

Usage

```
plot_ragas_means(summary_df, output_path, width = 1200, height = 700)
```

Arguments

- summary_df Output of `summarize_ragas()`.
- output_path Where to save the PNG.
- width, height Plot size in pixels.

Value

Invisibly, output_path.

qa_log_empty	<i>Create an empty QA log tibble</i>
--------------	--------------------------------------

Description

Standard schema for storing Q/A interactions produced by the RAG pipeline.

Usage

```
qa_log_empty()
```

Value

A tibble with zero rows and the standard QA log columns.

qa_metrics_empty	Create an empty QA metrics tibble
------------------	-----------------------------------

Description

This helper creates an empty tibble with the columns used to store RAGAS-style evaluation metrics for each QA interaction.

Usage

```
qa_metrics_empty()
```

Details

All metric values are expected to be in $[0, 1]$.

Value

A tibble with zero rows and the standard QA metrics columns.

query_rag	Query the RAG pipeline
-----------	------------------------

Description

Takes a user question, embeds it, retrieves relevant chunks from a vector store, constructs a grounded RAG prompt, and calls an LLM to generate an answer.

Usage

```
query_rag(  
  question,  
  collection = "default",  
  top_k = 4L,  
  embedding_model = "text-embedding-3-small",  
  chat_model = "gpt-4o-mini",  
  temperature = 0,  
  max_output_tokens = NULL,  
  score_threshold = 0,  
  system_prompt = ""  
)
```

Arguments

question	Character scalar.
collection	Vector store collection name.
top_k	Integer; number of chunks to retrieve.
embedding_model	Character; embedding model for the question.
chat_model	Character; chat model for answer generation.
temperature	Numeric; sampling temperature for the chat model.
max_output_tokens	Integer or NULL; maximum output tokens for the chat model.
score_threshold	Numeric; minimum similarity score to keep a retrieved chunk. Applied only if retrieved includes a score column.
system_prompt	Character; API-level system instruction passed to the chat model.

Value

A list with:

answer	Model-generated answer
retrieved	Tibble/data frame of retrieved chunks after filtering
prompt	Final grounded RAG prompt sent as the user/content prompt
model	Chat model used

save_qa_log	<i>Save a QA log to disk</i>
-------------	------------------------------

Description

This helper saves a QA log tibble, as produced by `log_rag_interaction()`, to an RDS file. By default, it writes to `db/qa_log.rds` inside the project directory.

Usage

```
save_qa_log(qa_log, path = "db/qa_log.rds")
```

Arguments

qa_log	A tibble created by <code>qa_log_empty()</code> and populated by <code>log_rag_interaction()</code> .
path	Character scalar; path to the RDS file. Defaults to <code>"db/qa_log.rds"</code> .

Value

Invisibly, the path to the saved file.

save_qa_metrics	<i>Save QA metrics to disk</i>
-----------------	--------------------------------

Description

This helper saves a QA metrics tibble, as produced by `compute_ragas_metrics()`, to an RDS file. By default, it writes to `db/qa_metrics.rds` inside the project directory.

Usage

```
save_qa_metrics(qa_metrics, path = "db/qa_metrics.rds")
```

Arguments

qa_metrics	A tibble created by <code>compute_ragas_metrics()</code> .
path	Character scalar; path to the RDS file. Defaults to <code>"db/qa_metrics.rds"</code> .

Value

Invisibly, the path to the saved file.

summarize_ragas	<i>Summarize RAGAS metrics across QA pairs</i>
-----------------	--

Description

Computes mean, standard deviation, minimum, and maximum for each metric column in a QA-metrics tibble.

Usage

```
summarize_ragas(qa_metrics)
```

Arguments

qa_metrics	A tibble returned by <code>compute_ragas_metrics()</code> or <code>compute_ragas_metrics_llm()</code> with metric columns such as <code>context_precision</code> , <code>context_recall</code> , <code>answer_relevance</code> , <code>faithfulness</code> , and <code>ragas_overall</code> .
------------	---

Value

A tibble with columns: `metric`, `mean`, `sd`, `min`, `max`.

`vectorstore_clear_all` *Delete ALL collections from the vector store*

Description

Completely wipes the vector store by replacing it with an empty tibble.

Usage

```
vectorstore_clear_all()
```

Value

Invisibly, TRUE.

`vectorstore_delete_collection`
Delete a collection from the R-native vector store

Description

Delete a collection from the R-native vector store

Usage

```
vectorstore_delete_collection(collection)
```

Arguments

`collection` Character; collection name to delete.

Value

Invisibly, TRUE on success.

vectorstore_load	<i>Load the R-native vector store</i>
------------------	---------------------------------------

Description

This function loads the current contents of the R-native vector store from `db/vectorstore.rds`. If the file does not exist yet, it returns an empty tibble with the expected columns: `collection`, `id`, `text`, `embedding`, and `metadata`.

Usage

```
vectorstore_load()
```

Value

A tibble with one row per stored chunk, or zero rows if the store is empty.

vectorstore_query	<i>Query the R-native vector store for nearest neighbors</i>
-------------------	--

Description

Query the R-native vector store for nearest neighbors

Usage

```
vectorstore_query(collection, query_embedding, top_k = 4L)
```

Arguments

<code>collection</code>	Character; name of the collection.
<code>query_embedding</code>	Numeric vector representing the query.
<code>top_k</code>	Integer; number of neighbors to retrieve.

Value

A tibble with columns `collection`, `id`, `text`, `score`, and `metadata`.

vectorstore_upsert	<i>Upsert embeddings into the R-native vector store</i>
--------------------	---

Description

Upsert embeddings into the R-native vector store

Usage

```
vectorstore_upsert(collection, ids, embeddings, documents, metadatas = NULL)
```

Arguments

collection	Character; name of the collection.
ids	Character vector of IDs, one for each embedding.
embeddings	Numeric matrix or data frame; one row per id.
documents	Character vector of raw text for each embedding.
metadatas	Optional list or data frame of metadata per row.

Value

Invisibly, TRUE on success.

Index

- * **NLP**
 - ragR-package, 2
- * **OpenAI**
 - ragR-package, 2
- * **RAGAS**
 - ragR-package, 2
- * **RAG**
 - ragR-package, 2
- * **chatbot**
 - ragR-package, 2
- * **embeddings**
 - ragR-package, 2
- api_chat_handler, 5
- api_clear_all_handler, 5
- api_clear_all_handler(), 4
- api_clear_handler, 6
- api_ragas_clear_handler, 6
- api_ragas_handler, 7
- api_ragas_report_handler, 7
- chunk_text_sentence, 8
- clear_qa_log, 8
- clear_qa_metrics, 9
- compute_ragas_metrics, 9
- compute_ragas_metrics(), 21
- compute_ragas_metrics_llm, 10
- compute_ragas_metrics_llm(), 21
- generate_openai_chat, 11
- generate_ragas_report, 12
- generate_ragas_report(), 12
- generate_ragas_report_llm, 12
- get_openai_embeddings, 13
- ingest_documents, 14
- load_qa_log, 15
- load_qa_metrics, 16
- load_rag_config, 16
- log_rag_interaction, 17
- log_rag_interaction(), 10, 20
- plot_ragas_means, 18
- qa_log_empty, 18
- qa_log_empty(), 8, 15, 20
- qa_metrics_empty, 19
- qa_metrics_empty(), 9, 16
- query_rag, 19
- query_rag(), 17
- ragR (ragR-package), 2
- ragR-package, 2
- save_qa_log, 20
- save_qa_metrics, 21
- summarize_ragas, 21
- summarize_ragas(), 18
- vectorstore_clear_all, 22
- vectorstore_clear_all(), 4, 5
- vectorstore_delete_collection, 22
- vectorstore_load, 23
- vectorstore_query, 23
- vectorstore_upsert, 24