Package: certellm (via r-universe)

May 16, 2026

Title A Certe R Package for working with large language models

Version 0.2.4

Description A Certe R Package for working with large language models.
This package is part of the 'certedata' universe.

URL https://certe-medical-epidemiology.github.io/certellm,
https://github.com/certe-medical-epidemiology/certellm

Depends R (>= 4.2.0)

Imports cli (>= 3.6.0), ellmer (>= 0.4.0), yaml

Suggests rlang, plot2, testthat (>= 2.0.0)

Remotes msberends/plot2

License GPL-2

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/testthat/edition 2

Config/pak/sysreqs libssl-dev

Repository https://certe-medical-epidemiology.r-universe.dev

Date/Publication 2026-04-16 06:26:41 UTC

RemoteUrl https://github.com/certe-medical-epidemiology/certellm

RemoteRef HEAD

RemoteSha 3e5a52f6a1abb63e542e0b86494a5ac3774721f4

Contents

`analyse` *Data Analysis and Code Assistance*

Description

These functions use the active LLM to assist with data analysis and R code tasks. Each call uses an isolated chat context. Generated code is returned as a character string for the user to review and run manually.

- `llm_analyse()` suggests analyses or answers a question about a dataset.
- `llm_code()` generates R code for a described task.
- `llm_explain()` explains R code in Dutch.
- `llm_suggest_plot()` suggests a `plot2()` call for a dataset.

Usage

```
llm_analyse(data, question = NULL, new = FALSE, ...)

llm_code(data = NULL, task, style = c("tidyverse", "base"), new = FALSE, ...)

llm_explain(code, detail = c("brief", "detailed"), new = FALSE, ...)

llm_suggest_plot(data, goal = NULL, new = FALSE, ...)
```

Arguments

`data`	A data frame. Summarised before sending to the LLM; raw rows are never sent verbatim.
`question`	A specific question about the data. If `NULL`, the LLM is asked to suggest appropriate analyses.
`new`	Logical. If `TRUE`, re-initiates the LLM before the task call. Default is `FALSE`.
`...`	Additional arguments passed to `initiate_llm()`.
`task`	Description of what the R code should accomplish.
`style`	Coding style for generated R code: `"tidyverse"` (default) or `"base"`.
`code`	A character string containing R code to explain or improve.
`detail`	Level of detail for explanations: `"brief"` (one paragraph, default) or `"detailed"` (step-by-step).
`goal`	Optional description of what the plot should show.

Value

A character string with the LLM response (suggestions, code, or explanation).

chat *Chat With (Local) LLM*

Description

These functions initialise and interact with the active LLM, backed by `ellmer::chat()`. The connection is configured via named presets defined in the secrets YAML under `llm.presets`; individual `provider`, `model`, and `url` arguments override the preset when supplied.

Usage

```
initiate_llm(
  preset = read_secret("llm.default.preset"),
  provider = NULL,
  model = NULL,
  url = NULL,
  system_prompt = read_secret("llm.system_prompt"),
  user_name = read_secret(paste0("user.", Sys.info()["user"], ".fullname")),
  user_jobtitle = read_secret(paste0("user.", Sys.info()["user"], ".jobtitle")),
  ...
)

list_presets()

new_chat(input, ...)

chat(input, new = FALSE, ...)

chat_in_browser(new = FALSE, ...)

chat_in_console(new = FALSE, ...)

get_chat_object()

get_provider()

get_system_prompt()

get_tools()

get_tokens()
```

Arguments

`preset`	Name of a preset defined under `llm.presets` in the secrets YAML. Defaults to `llm.default.preset`. Each preset must contain a `provider` (e.g. `"ollama"`, `"anthropic"`) and a `model` field; an optional `url` sets a non-default base URL

	(e.g. an internal Ollama server). Individual `provider`, `model`, and `url` arguments override the preset when supplied.
`provider`	Override the provider from the preset.
`model`	Override the model from the preset.
`url`	Override the base URL from the preset. Passed as `base_url` to the provider function; omit for providers with fixed endpoints (e.g. Anthropic).
`system_prompt`	System prompt text. Defaults to `llm.system_prompt` from the secrets file.
`user_name, user_jobtitle`	Name and job title of the person working with the LLM.
`...`	Arguments passed on to `ellmer::chat()`.
`input`	Text for input.
`new`	Initiate new LLM instance.

Details

Functions:

- `initiate_llm()` creates a new `ellmer` Chat object, appends user context to the system prompt when the user's name and job title are found in the secrets file, registers the session tools (`get_df_summary`, `list_objects`, `get_colnames`), and probes the model to verify actual tool support. Models that do not support tools are silently recreated without them, so the function always leaves a working instance. The active object is stored in `pkg_env$chat_object`.
- `list_presets()` prints all configured presets with their provider, model, and URL, marking the default.
- `chat()` sends a single message to the active LLM, initiating a new session first if none exists or if `new = TRUE`.
- `new_chat()` is a shorthand that always initiates a fresh session before sending the message.
- `chat_in_browser()` and `chat_in_console()` open an interactive session via `ellmer::live_browser()` and `ellmer::live_console()` respectively.
- `get_chat_object()`, `get_provider()`, `get_system_prompt()`, `get_tools()`, and `get_tokens()` are thin accessors that expose the underlying `ellmer` Chat object's state for inspection.

`chat_history` *Save and Restore Chat Conversations*

Description

These functions manage the turn history of the active LLM conversation so that sessions can be saved, restored, listed, or reset.

- `save_chat()` snapshots the current conversation under a name.
- `restore_chat()` replaces the current conversation with a saved snapshot.
- `list_chats()` returns the names of all saved snapshots.
- `reset_chat()` clears the current conversation turn history.

Usage

```
save_chat(name = format(Sys.time(), "%Y%m%d_%H%M%S"))

restore_chat(name)

list_chats()

reset_chat()
```

Arguments

name	Name under which to save or from which to restore the snapshot. Defaults to a timestamp ("YYYYMMDD_HHMMSS").

Value

save_chat() and reset_chat() return invisible(NULL). restore_chat() returns invisible(NULL). list_chats() returns a character vector of snapshot names (or an empty vector if none exist).

---

diver *Diver Database Query Assistance*

Description

These functions help construct and explain get_diver_data() calls (from the certedb package) using the active LLM. Each call uses an isolated chat context so the main conversation history is not affected.

- llm_diver_query() generates a get_diver_data() call from a natural language description.
- llm_diver_explain() explains an existing get_diver_data() call in plain Dutch.

The LLM is primed with a static schema hint describing known Diver column names and conventions. When the exact column name is uncertain, the generated code includes a comment noting the assumption.

Usage

```
llm_diver_query(
  description,
  table = NULL,
  columns_hint = NULL,
  new = FALSE,
  ...
)

llm_diver_explain(query, new = FALSE, ...)
```

Arguments

description	Natural language description of the data to retrieve, e.g. "All E. coli urine cultures from 2024 in Groningen".
table	Optional hint for the Diver table to query.
columns_hint	Optional character vector of known column names to assist the LLM in forming the query.
new	Logical. If TRUE, re-initiates the LLM before the task call. Default is FALSE.
...	Additional arguments passed to initiate_llm().
query	An existing get_diver_data() call as a character string, to be explained.

Value

A character string: the generated R code (llm_diver_query()) or the Dutch explanation (llm_diver_explain()).

---

document *Rmd / Quarto Document Assistance*

Description

These functions use the active LLM to assist with writing and reviewing R Markdown and Quarto documents. Each call uses an isolated chat context.

- llm_review_document() reviews a document for language, structure, or completeness and returns Dutch-language feedback.
- llm_write_section() writes a Dutch section (e.g. Methods, Results) for a report, presentation, or article.
- llm_write_chunk() generates or improves an R code chunk for a document.

Usage

```
llm_review_document(
  file = NULL,
  text = NULL,
  focus = c("all", "language", "structure", "completeness"),
  new = FALSE,
  ...
)

llm_write_section(
  section,
  data = NULL,
  context = NULL,
  style = c("report", "presentation", "article"),
  new = FALSE,
  ...
```

```
)

llm_write_chunk(code = NULL, goal = NULL, new = FALSE, ...)
```

Arguments

file	Path to an .Rmd or .qmd file. The first 3000 lines are read. Either file or text must be supplied.
text	The document content as a character string. Either file or text must be supplied.
focus	What to focus on during review: "language", "structure", "completeness", or "all" (default).
new	Logical. If TRUE, re-initiates the LLM before the task call. Default is FALSE.
...	Additional arguments passed to the LLM chat method.
section	Name of the section to write, e.g. "Methods", "Results", "Discussion".
data	Optional data frame providing context for the section. Summarised before sending to the LLM.
context	Optional character string with additional background context for the section (e.g. a description of the analysis).
style	Document style: "report" (default), "presentation", or "article".
code	Optional existing R code to improve (as a character string).
goal	Description of what the code chunk should accomplish. Required when code is NULL.

Value

A character string with the generated or reviewed text.

---

narrative *Generate Dutch Epidemiological Narratives*

Description

These functions use the active LLM to produce Dutch-language text based on data from the Medical Epidemiology department. Each call uses an isolated chat context so the main conversation history is not affected.

- `llm_describe()` writes a Dutch epidemiological description of a dataset.
- `llm_interpret()` interprets patterns and suggests epidemiological explanations.
- `llm_conclude()` writes a Dutch conclusion paragraph from results or data.

Usage

```
llm_describe(
  data,
  topic = NULL,
  period = NULL,
  region = NULL,
  tone = c("formal", "accessible"),
  length = c("short", "medium", "long"),
  new = FALSE,
  ...
)

llm_interpret(data, hypothesis = NULL, comparison = NULL, new = FALSE, ...)

llm_conclude(..., new = FALSE)
```

Arguments

data	A data frame to describe or interpret. Summarised before sending to the LLM; raw rows are never sent verbatim.
topic	Topic or subject of the analysis, e.g. "E. coli resistance".
period	Time period covered, e.g. "2023-2024".
region	Geographic region, e.g. "Noord-Nederland" or "Groningen".
tone	Writing tone: "formal" (default) or "accessible".
length	Approximate length of the output: "short" (1 paragraph), "medium" (2-3 paragraphs), or "long" (4+ paragraphs).
new	Logical. If TRUE, re-initiates the LLM before the task call. Default is FALSE.
...	Additional arguments passed to initiate_llm().
hypothesis	Optional hypothesis to evaluate against the data.
comparison	Optional second data frame for comparison. Also summarised before sending.

Value

A character string with the generated Dutch text.

Index