

# Package ‘llmhelper’

May 8, 2026

**Type** Package

**Title** Unified Interface for Large Language Model Interactions

**Version** 1.0.0

**Description** Provides a unified interface for interacting with Large Language Models (LLMs) through various providers including OpenAI <<https://platform.openai.com/docs/api-reference>>, Ollama <<https://ollama.com/>>, and other OpenAI-compatible APIs. Features include automatic connection testing, max\_tokens limit auto-adjustment, structured JSON responses with schema validation, interactive JSON schema generation, prompt templating, and comprehensive diagnostics.

**License** GPL (>= 3)

**URL** <https://github.com/Zaoqu-Liu/llmhelper>

**BugReports** <https://github.com/Zaoqu-Liu/llmhelper/issues>

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Depends** R (>= 4.1.0)

**Imports** cli, dplyr, glue, httr, httr2, jsonlite, purrr, stringr, tibble, tidyprompt

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Zaoqu Liu [aut, cre] (ORCID: <<https://orcid.org/0000-0002-0452-742X>>)

**Maintainer** Zaoqu Liu <liuzaoqu@163.com>

**Repository** CRAN

**Date/Publication** 2026-01-31 18:50:07 UTC

## Contents

build_prompt . . . . .	2
diagnose_llm_connection . . . . .	3
extract_schema_only . . . . .	4
generate_json_schema . . . . .	4
get_llm_response . . . . .	5
get_user_feedback . . . . .	9
grapes-or-or-grapes . . . . .	9
llm_ollama . . . . .	10
llm_provider . . . . .	11
ollama_delete_model . . . . .	12
ollama_download_model . . . . .	13
ollama_list_models . . . . .	13
set_prompt . . . . .	14

<b>Index</b>	<b>15</b>
--------------	-----------

---

build_prompt	<i>Build a templated prompt for LLM interaction using glue</i>
--------------	--

---

### Description

This function constructs a structured prompt string by injecting user-supplied parameters into a predefined template. It leverages the glue package to replace named placeholders in the template with actual values, enabling dynamic prompt creation for LLM workflows.

### Usage

```
build_prompt(template, ...)
```

### Arguments

template	A character string containing the prompt template. Placeholders should be wrapped in {} and correspond to names provided in ....
...	Named arguments matching placeholders in template. Each name–value pair will be substituted into the template at runtime.

### Details

The build\_prompt() function uses glue::glue\_data() internally. Placeholders in template (e.g., {filename}, {threshold}) are resolved by passing a named list of parameters via .... You can include any number of placeholders in the template, as long as the corresponding argument is supplied when calling this function.

### Value

A single character string with all {placeholder} fields in template replaced by the corresponding values from ....

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

**Examples**

```
## Not run:
# Define a template with placeholders
prompt_template <- "
Perform the following analysis on dataset at '{filepath}':
1. Load data from '{filepath}'
2. Normalize using method '{norm_method}'
3. Save results to '{output_dir}'

IMPORTANT: Use package::function notation for all function calls."

# Build the prompt by supplying named arguments
filled_prompt <- build_prompt(
  template      = prompt_template,
  filepath      = "/path/to/data.csv",
  norm_method   = "quantile",
  output_dir    = "/path/to/output/"
)
cat(filled_prompt)

## End(Not run)
```

---

diagnose\_llm\_connection

*Comprehensive LLM connection diagnostics*

---

**Description**

This function provides detailed diagnostics for LLM connection issues, helping identify problems at different levels of the stack.

**Usage**

```
diagnose_llm_connection(base_url, api_key, model, test_tidyprompt = TRUE)
```

**Arguments**

base_url	The API base URL
api_key	The API key
model	The model name
test_tidyprompt	Whether to test tidyprompt compatibility

**Value**

A list (invisible) containing test results with elements:

- connectivity: Logical indicating if basic network connectivity passed
- endpoint: Logical indicating if API endpoint is accessible
- auth: Logical indicating if authentication and model test passed
- tidyprompt: Logical indicating tidyprompt compatibility (if tested)

---

`extract_schema_only`    *Extract only the schema part from generated result*

---

**Description**

Extract only the schema part from generated result

**Usage**

```
extract_schema_only(schema_result)
```

**Arguments**

`schema_result`    Result from `generate_json_schema`

**Value**

Just the schema portion for use with tidyprompt

---

`generate_json_schema`    *Interactive JSON Schema Generator using tidyprompt*

---

**Description**

This function creates an interactive system to generate JSON schemas based on user descriptions. It supports multi-turn conversations until the user is satisfied with the generated schema.

**Usage**

```
generate_json_schema(  
    description,  
    llm_client,  
    max_iterations = 5,  
    interactive = TRUE,  
    verbose = TRUE  
)
```

**Arguments**

description	Initial description of the desired JSON structure
llm_client	The LLM provider object (from llm_openai or llm_ollama)
max_iterations	Maximum number of refinement iterations (default: 5)
interactive	Whether to run in interactive mode (default: TRUE)
verbose	Whether to show detailed conversation logs (default: TRUE)

**Value**

A list containing the final JSON schema and conversation history

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

---

get\_llm\_response      *Get LLM Response with Text or JSON Output*

---

**Description**

This function sends a prompt to a Language Learning Model (LLM) and returns either a text response or a JSON-structured response based on the provided parameters. It handles retries, validation, and response formatting automatically.

**Usage**

```
get_llm_response(  
  prompt,  
  llm_client,  
  max_retries = 5,  
  max_words = NULL,  
  max_characters = NULL,  
  json_schema = NULL,  
  schema_strict = FALSE,  
  schema_type = "auto",  
  verbose = NULL,  
  stream = NULL,  
  clean_chat_history = TRUE,  
  return_mode = c("only_response", "full")  
)
```

**Arguments**

prompt	A character string or tidyprompt object containing the prompt to send to the LLM. This is the main input that the LLM will respond to.
llm_client	An LLM provider object created by functions like llm_openai() or llm_ollama(). This object contains the configuration for connecting to and communicating with the specific LLM service.
max_retries	Integer. Maximum number of retry attempts if the LLM fails to provide a valid response (default: 5). The function will retry if: <ul style="list-style-type: none"> <li>• The response doesn't meet validation criteria</li> <li>• JSON parsing fails (when using json_schema)</li> <li>• Network or API errors occur If max_retries is exceeded, NULL is returned.</li> </ul>
max_words	Integer or NULL. Maximum number of words allowed in the response (default: NULL, no limit). Only applies when json_schema is NULL (text responses). If specified, responses exceeding this limit will trigger a retry. Example: max_words = 50 limits response to 50 words or fewer.
max_characters	Integer or NULL. Maximum number of characters allowed in the response (default: NULL, no limit). Only applies when json_schema is NULL (text responses). If specified, responses exceeding this limit will trigger a retry. Example: max_characters = 280 limits response to Twitter-like length.
json_schema	List or NULL. JSON schema specification for structured responses (default: NULL for text responses). When provided, the LLM will be forced to return a valid JSON object matching the schema. The schema should be a list representing a JSON schema structure with: <ul style="list-style-type: none"> <li>• name: Schema identifier</li> <li>• description: Schema description</li> <li>• schema: The actual JSON schema with type, properties, required fields, etc. Example: list(name = "person", schema = list(type = "object", properties = ...))</li> </ul>
schema_strict	Logical. Whether to enforce strict schema validation (default: FALSE). When TRUE: <ul style="list-style-type: none"> <li>• JSON responses must exactly match the schema</li> <li>• No additional properties are allowed beyond those specified</li> <li>• All required fields must be present Only applicable when json_schema is provided.</li> </ul>
schema_type	Character. Method for enforcing JSON response format (default: 'auto'). Options: <ul style="list-style-type: none"> <li>• 'auto': Automatically detect best method based on LLM provider</li> <li>• 'text-based': Add JSON instructions to prompt (works with any provider)</li> <li>• 'openai': Use OpenAI's native JSON mode (requires compatible OpenAI API)</li> <li>• 'ollama': Use Ollama's native JSON mode (requires compatible Ollama model)</li> <li>• 'openai_oo': OpenAI mode without schema enforcement in API</li> </ul>

	<ul style="list-style-type: none"> <li>• 'ollama_oo': Ollama mode without schema enforcement in API</li> </ul>
verbose	<p>Logical or NULL. Whether to print detailed interaction logs to console (default: NULL, uses LLM client's setting). When TRUE:</p> <ul style="list-style-type: none"> <li>• Shows the prompt being sent</li> <li>• Displays the LLM's response</li> <li>• Reports retry attempts and validation failures Useful for debugging and monitoring LLM interactions.</li> </ul>
stream	<p>Logical or NULL. Whether to stream the response in real-time (default: NULL, uses LLM client's setting). When TRUE:</p> <ul style="list-style-type: none"> <li>• Response appears progressively as the LLM generates it</li> <li>• Provides faster perceived response time</li> <li>• Only works if the LLM provider supports streaming Note: Streaming is automatically disabled when verbose = FALSE.</li> </ul>
clean_chat_history	<p>Logical. Whether to clean chat history between retries (default: TRUE). When TRUE:</p> <ul style="list-style-type: none"> <li>• Keeps only essential messages in context (first/last user message, last assistant message, system messages)</li> <li>• Reduces context window usage on retries</li> <li>• May improve performance with repeatedly failing responses When FALSE, full conversation history is maintained.</li> </ul>
return_mode	<p>Character. What information to return (default: "only_response"). Options:</p> <ul style="list-style-type: none"> <li>• "only_response": Returns only the processed LLM response (character string or parsed JSON)</li> <li>• "full": Returns a comprehensive list containing: <ul style="list-style-type: none"> <li>– response: The processed LLM response</li> <li>– interactions: Number of interactions with the LLM</li> <li>– chat_history: Complete conversation history</li> <li>– chat_history_clean: Cleaned conversation history</li> <li>– start_time: When the function started</li> <li>– end_time: When the function completed</li> <li>– duration_seconds: Total execution time</li> <li>– http_list: Raw HTTP responses from the API</li> </ul> </li> </ul>

## Details

This function serves as a unified interface for getting responses from LLMs with automatic handling of different response formats and validation. It internally uses the tidyprompt package's `answer_as_text()` or `answer_as_json()` functions depending on whether a JSON schema is provided.

### Text Mode (`json_schema = NULL`):

- Uses `answer_as_text()` with optional word/character limits
- Returns plain text responses

- Validates response length constraints

#### **JSON Mode (json\_schema provided):**

- Uses answer\_as\_json() with schema validation
- Forces structured JSON responses
- Validates against provided schema
- Returns parsed R objects (lists)

**Error Handling:** The function automatically retries on various failure conditions including validation errors, JSON parsing errors, and network issues.

#### **Value**

Depends on return\_mode parameter:

- If return\_mode = "only\_response": Character string (text mode) or parsed list (JSON mode)
- If return\_mode = "full": Named list with response and metadata
- NULL if all retry attempts fail

#### **Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

#### **Examples**

```
## Not run:
# Basic text response
client <- llm_ollama()
response <- get_llm_response("What is R?", client)

# Text response with word limit
short_response <- get_llm_response(
  "Explain machine learning",
  client,
  max_words = 50
)

# JSON response with schema
schema <- list(
  name = "person_info",
  schema = list(
    type = "object",
    properties = list(
      name = list(type = "string"),
      age = list(type = "integer")
    ),
    required = c("name", "age")
  )
)
```

```
json_response <- get_llm_response(  
  "Create a person with name and age",  
  client,  
  json_schema = schema  
)  
  
# Full response with metadata  
full_result <- get_llm_response(  
  "Hello",  
  client,  
  return_mode = "full",  
  verbose = TRUE  
)  
  
## End(Not run)
```

---

get\_user\_feedback      *Get user feedback interactively*

---

### Description

Get user feedback interactively

### Usage

```
get_user_feedback(state, verbose)
```

### Arguments

state	Current conversation state
verbose	Show logs

### Value

Updated conversation state

---

grapes-or-or-grapes      *Null coalescing operator*

---

### Description

Returns the left-hand side if it is not NULL, otherwise returns the right-hand side. This is useful for providing default values.

**Usage**

```
x %||% y
```

**Arguments**

```
x          A value to check for NULL.
y          A default value to return if x is NULL.
```

**Value**

x if not NULL, otherwise y.

**Examples**

```
NULL %||% "default"
"value" %||% "default"
```

---

llm_ollama	<i>Create Ollama LLM provider with enhanced availability check and auto-download</i>
------------	--

---

**Description**

This function creates an Ollama LLM provider with better error handling and follows tidyprompt best practices.

**Usage**

```
llm_ollama(
  base_url = "http://localhost:11434/api/chat",
  model = "qwen2.5:1.5b-instruct",
  temperature = 0.2,
  max_tokens = 5000,
  timeout = 100,
  stream = TRUE,
  verbose = TRUE,
  skip_test = FALSE,
  auto_download = TRUE,
  ...
)
```

**Arguments**

```
base_url    The base URL for the Ollama API
model       The model name to use
temperature The temperature parameter for response randomness
```

max_tokens	Maximum number of tokens in response
timeout	Request timeout in seconds
stream	Whether to use streaming responses
verbose	Whether to show verbose output
skip_test	Whether to skip the availability test
auto_download	Whether to automatically download missing models
...	Additional parameters to pass to the model

**Value**

A configured LLM provider object

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

---

llm_provider	<i>Create OpenAI-compatible LLM provider with enhanced error handling</i>
--------------	---

---

**Description**

This function creates an OpenAI-compatible LLM provider with comprehensive error handling and testing capabilities. It automatically handles max\_tokens limits by falling back to the model's maximum when exceeded.

**Usage**

```
llm_provider(
  base_url = "https://api.openai.com/v1/chat/completions",
  api_key = NULL,
  model = "gpt-4o-mini",
  temperature = 0.2,
  max_tokens = 5000,
  timeout = 100,
  stream = FALSE,
  verbose = TRUE,
  skip_test = FALSE,
  test_mode = c("full", "http_only", "skip"),
  ...
)
```

**Arguments**

base_url	The base URL for the OpenAI-compatible API
api_key	The API key for authentication. If NULL, will use LLM_API_KEY env var
model	The model name to use
temperature	The temperature parameter for response randomness
max_tokens	Maximum number of tokens in response (will auto-adjust if exceeds model limit)
timeout	Request timeout in seconds
stream	Whether to use streaming responses
verbose	Whether to show verbose output
skip_test	Whether to skip the availability test (useful for problematic providers)
test_mode	The testing mode: "full", "http_only", "skip"
...	Additional parameters to pass to the model

**Value**

A configured LLM provider object

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

---

ollama\_delete\_model *Delete a model from Ollama API*

---

**Description**

This function sends a DELETE request to remove a specified model from the Ollama API and returns the updated model list.

**Usage**

```
ollama_delete_model(.model, .ollama_server = "http://localhost:11434")
```

**Arguments**

.model	The name of the model to delete
.ollama_server	The URL of the Ollama server (default: "http://localhost:11434")

**Value**

Updated tibble of available models after deletion

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

---

ollama\_download\_model *Download a model from Ollama API*

---

**Description**

This function sends a request to download a specified model from Ollama's model library with progress tracking.

**Usage**

```
ollama_download_model(.model, .ollama_server = "http://localhost:11434")
```

**Arguments**

`.model` The name of the model to download  
`.ollama_server` The URL of the Ollama server (default: "http://localhost:11434")

**Value**

No return value, called for side effects (downloads the model to the Ollama server with progress displayed in the console).

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

---

ollama\_list\_models *List available models from Ollama API*

---

**Description**

This function retrieves information about available models from the Ollama API and returns it as a tibble with simplified data extraction.

**Usage**

```
ollama_list_models(.ollama_server = "http://localhost:11434")
```

**Arguments**

`.ollama_server` The URL of the Ollama server (default: "http://localhost:11434")

**Value**

A tibble containing model information, or NULL if no models are found

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

---

set\_prompt

*Set system and user prompts for LLM interaction*

---

**Description**

This function creates a prompt object with system and user prompts using the tidyprompt package for structured LLM communication.

**Usage**

```
set_prompt(  
  system = "You are an AI assistant specialized in bioinformatics.",  
  user = "Hi"  
)
```

**Arguments**

system	The system prompt to set context and behavior (default: bioinformatics assistant)
user	The user prompt or question

**Value**

A prompt object configured with system and user prompts

**Author(s)**

Zaoqu Liu; Email: liuzaoqu@163.com

# Index

[build\\_prompt](#), [2](#)

[diagnose\\_llm\\_connection](#), [3](#)

[extract\\_schema\\_only](#), [4](#)

[generate\\_json\\_schema](#), [4](#)

[get\\_llm\\_response](#), [5](#)

[get\\_user\\_feedback](#), [9](#)

[grapes-or-or-grapes](#), [9](#)

[llm\\_ollama](#), [10](#)

[llm\\_provider](#), [11](#)

[ollama\\_delete\\_model](#), [12](#)

[ollama\\_download\\_model](#), [13](#)

[ollama\\_list\\_models](#), [13](#)

[set\\_prompt](#), [14](#)