

# Package: llmjoin (via r-universe)

June 16, 2026

**Type** Package

**Title** LLM-Powered Fuzzy Join

**Version** 0.3.0

**Description** Resolves ambiguous links between data.frames using large language models (LLMs). Supports matching across spelling variations, translations, and differing levels of precision.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Config/testthat/edition** 3

**Imports** httr, jsonlite, config, readr

**Suggests** testthat (>= 3.0.0)

**Depends** R (>= 4.2.0)

**URL** <https://github.com/evanliu3594/llmjoin>

**BugReports** <https://github.com/evanliu3594/llmjoin/issues>

**Config/roxygen2/version** 8.0.0

**NeedsCompilation** no

**Author** Yifan LIU [aut, cre]

**Maintainer** Yifan LIU <yifan.liu@smail.nju.edu.cn>

**Repository** <https://cran.r-universe.dev>

**Date/Publication** 2026-06-16 19:50:02 UTC

**RemoteUrl** <https://github.com/cran/llmjoin>

**RemoteRef** HEAD

**RemoteSha** fd3d72987bd1a4b1042d380cc8a3102fa1961730

## Contents

|                        |   |
|------------------------|---|
| build_joint . . . . .  | 2 |
| chat_llm . . . . .     | 3 |
| joint_prompt . . . . . | 4 |
| llm_join . . . . .     | 4 |
| parse_joint . . . . .  | 5 |
| set_llm . . . . .      | 6 |
| tbl2md . . . . .       | 6 |

|              |          |
|--------------|----------|
| <b>Index</b> | <b>8</b> |
|--------------|----------|

---

|             |  |
|-------------|--|
| build_joint | <i>Build a fuzzy-join joint data.frame via LLM</i> |
|-------------|--|

---

### Description

Build a fuzzy-join joint data.frame via LLM

### Usage

```
build_joint(x, y, key1, key2, ...)
```

### Arguments

|      |   |
|------|---|
| x    | a data.frame to be joined on the lhs.                               |
| y    | a data.frame to be joined on the rhs.                               |
| key1 | string, name of the key column of data.frame x waiting for pairing. |
| key2 | string, name of the key column of data.frame y waiting for pairing. |
| ...  | extra params passed to chat_llm()                                   |

### Value

a 2-column data.frame mapping values from key1 to key2.

### Examples

```
build_joint(
  x = data.frame(x = c("01", "02", "04")),
  y = data.frame(y = c("January", "Feb", "May")),
  key1 = "x", key2 = "y"
)
```

---

|          |                                   |
|----------|-----------------------------------|
| chat_llm | <i>Send message to LLM server</i> |
|----------|-----------------------------------|

---

## Description

This function sends a message to the LLM model and retrieves the result.

## Usage

```
chat_llm(  
  .message,  
  .model = NULL,  
  .temperature = 0,  
  .max_tokens = 30000,  
  .timeout = 300,  
  .verbose = getOption("llmjoin.verbose", FALSE)  
)
```

## Arguments

|              |  |
|--------------|--|
| .message     | the message to send.   |
| .model       | character, LLM model to use. By default NULL (uses config value).                            |
| .temperature | OpenAI style randomness control (0~1), by default 0.   |
| .max_tokens  | Max tokens to spend.   |
| .timeout     | Max seconds to communicate with LLM.   |
| .verbose     | logical, print progress messages. Default <code>getOption("llmjoin.verbose", FALSE)</code> . |

## Value

A character string with the LLM's response text.

## Examples

```
chat_llm("tell a joke.")
```

---

|              |                                  |
|--------------|----------------------------------|
| joint_prompt | <i>Generate connector prompt</i> |
|--------------|----------------------------------|

---

### Description

Generate a prompt to guide the LLM in generating a joint for data frame joining, leveraging the two key columns from the tables to be connected. As of 2025/04/10, DeepSeek R1 and gpt-4.1-mini showed the best result; other LLMs might fabricate non-existent data in the result.

### Usage

```
joint_prompt(x, y)
```

### Arguments

|   |  |
|---|--|
| x | 1-column data.frame or vector of characters, left hand side of the join  |
| y | 1-column data.frame or vector of characters, right hand side of the join |

### Value

A character string containing the matching prompt.

### Examples

```
joint_prompt(
  data.frame(x = c("01", "02", "04")),
  data.frame(y = c("January", "Feb", "May"))
)
```

---

|          |                            |
|----------|----------------------------|
| llm_join | <i>Fuzzy join with LLM</i> |
|----------|----------------------------|

---

### Description

Fuzzy join with LLM

### Usage

```
llm_join(x, y, key1, key2, ...)
```

### Arguments

|      |   |
|------|---|
| x    | a data.frame to be joined on the lhs.                               |
| y    | a data.frame to be joined on the rhs.                               |
| key1 | string, name of the key column of data.frame x waiting for pairing. |
| key2 | string, name of the key column of data.frame y waiting for pairing. |
| ...  | extra params passed to chat_llm()                                   |

**Value**

the fuzzy-joined data.frame

**Examples**

```
x <- data.frame(id = c("01", "02", "04"), value = c(10, 20, 40))
y <- data.frame(month = c("January", "Feb", "May"), amount = c(100, 200, 400))

llm_join(x, y, key1 = "id", key2 = "month")
```

---

parse\_joint

*Parse LLM response into a fuzzy-join joint data.frame*

---

**Description**

Strips markdown fences, extracts the longest consecutive block of comma-separated lines, ensures a header row matching 'key1,key2' is present, and parses the CSV into a 2-column data.frame.

**Usage**

```
parse_joint(llm_response, key1, key2)
```

**Arguments**

llm\_response    character, raw response from the LLM.  
key1            string, name of the lhs key column.  
key2            string, name of the rhs key column.

**Value**

a 2-column data.frame mapping values from key1 to key2.

**Examples**

```
parse_joint("01,January\n02,Feb\n04,May", key1 = "id", key2 = "month")
```

---

|         |                                |
|---------|--------------------------------|
| set_llm | <i>Set up your LLM service</i> |
|---------|--------------------------------|

---

### Description

Set up your LLM service with native support for OpenAI, Claude (Anthropic), and Gemini (via OpenAI-compatible endpoint). For custom endpoints like Ollama, proxies, DeepSeek, Kimi, and others, use provider = "openai" along with your custom URL to connect through the compactible API interface. All information is stored strictly locally in your system configuration and is never uploaded or shared.

### Usage

```
set_llm(provider = "openai", url = NULL, key = NULL, model = NULL)
```

### Arguments

|          |   |
|----------|---|
| provider | character, LLM provider. One of "openai", "claude", "gemini". Default "openai". |
| url      | url to your LLM provider endpoint. If NULL, auto-set based on provider.         |
| key      | api-key of your service.  |
| model    | character, model name. If NULL, auto-set from provider default.                 |

### Value

NULL invisibly. Called for side effect of writing the config file.

### Examples

```
set_llm(provider = "openai", key = "<your-openai-api-key>", model = "gpt-5.4-mini")
```

---

|        |   |
|--------|---|
| tbl2md | <i>Convert a data frame to a markdown table</i> |
|--------|---|

---

### Description

Convert a data frame to a markdown table

### Usage

```
tbl2md(tbl, nm = NULL)
```

**Arguments**

`tbl` a data.frame object or a vector.  
`nm` character, only used if 'tbl' is a vector.

**Value**

markdown style table string lines

**Examples**

```
tbl2md(iris)
```

# Index

`build_joint`, 2

`chat_llm`, 3

`joint_prompt`, 4

`llm_join`, 4

`parse_joint`, 5

`set_llm`, 6

`tbl2md`, 6