# Package: chatRater (via r-universe)

May 15, 2026

**Type** Package

**Title** A Tool for Rating Text/Image/Audio Stimuli via 'LLMs'

**Version** 1.3.1

**Date** 2026-05-15

**Maintainer** Shiyang Zheng <Shiyang.Zheng@nottingham.ac.uk>

**Description** Evaluates stimuli using Large Language Models. Supports multiple LLM providers: 'OpenAI', 'Anthropic', 'Ollama', 'LM Studio', 'DeepSeek', 'Groq', 'Mistral', and 'OpenAI-compatible' endpoints. Stimuli: plain text, local image/audio files, or image URLs. Audio is transcribed via 'OpenAI Whisper' before rating. Supports numeric, text, and raw return types.

**License** MIT + file LICENSE

**Depends** R (>= 4.1.0)

**Encoding** UTF-8

**Imports** base64enc, tools, httr2, curl, jsonlite

**Suggests** llmcoder (>= 1.2.0), testthat (>= 3.0.0)

**NeedsCompilation** no

**RoxygenNote** 7.3.2

**Config/testthat/edition** 3

**URL** https://github.com/ShiyangZheng/chatRater

**BugReports** https://github.com/ShiyangZheng/chatRater/issues

**Author** Shiyang Zheng [aut, cre]

**Config/pak/sysreqs** libssl-dev

**Repository** https://shiyangzheng.r-universe.dev

**Date/Publication** 2026-05-15 17:33:25 UTC

**RemoteUrl** https://github.com/shiyangzheng/chatrater

**RemoteRef** HEAD

**RemoteSha** 71b74d6898d1f661c76716ada3768b460edfaab9

# Contents



---

`alignment` *Validate LLM Ratings Against Human Norms*

---

## Description

Computes agreement statistics between two rating sources (e.g., LLM-generated ratings vs. human norms), including correlation analysis (Pearson, Spearman), paired t-tests for systematic bias, Cohen's d effect sizes, and Bland-Altman plots. Designed for the validation workflow described in Brysbaert et al. (2025).

## Usage

```
alignment(
  x,
  y,
  x_label = "Source 1",
  y_label = "Source 2",
  dim_name = "Rating",
  plot = TRUE,
  return_stats = TRUE
)
```

## Arguments

| | |
|---|---|
| `x` | Numeric vector of ratings from source 1 (e.g., LLM). |
| `y` | Numeric vector of ratings from source 2 (e.g., human norms). Must be the same length as `x`. |
| `x_label` | Label for source 1 (default: "Source 1"). |
| `y_label` | Label for source 2 (default: "Source 2"). |
| `dim_name` | Name of the rated dimension (default: "Rating"). Used in plot titles and console output. |
| `plot` | Logical. If `TRUE` (default), displays a correlation scatter plot and a Bland-Altman plot in the active graphics device. |
| `return_stats` | Logical. If `TRUE` (default), returns a list of alignment statistics invisibly. |

## Value

If `return_stats = TRUE`, a list with:

- `n` — number of paired observations
- `mean_x`, `mean_y` — group means
- `mean_diff`, `sd_diff` — mean and SD of differences
- `pearson_r`, `pearson_p`, `pearson_ci` — Pearson correlation
- `spearman_rho`, `spearman_p` — Spearman correlation
- `paired_t`, `paired_df`, `paired_p` — paired t-test
- `cohens_d` — Cohen's d for the mean difference
- `bland_altman_loa` — Bland-Altman 95% limits of agreement

## Examples

```
## Not run:
# Generate ratings with the weighted method, then validate
llm_ratings <- c(3.1, 4.5, 2.8, 3.9, 4.2)
human_ratings <- c(3.0, 4.8, 2.5, 4.0, 4.5)

result <- alignment(
  x        = llm_ratings,
  y        = human_ratings,
  x_label  = "GPT-4o",
  y_label  = "Human (LT2008)",
  dim_name = "Familiarity",
  plot     = TRUE
)

# Access returned statistics
print(result$pearson_r)   # e.g. 0.74
print(result$cohens_d)    # e.g. 0.32
print(result$paired_p)    # e.g. 0.001

## End(Not run)
```

---

`generate_ratings` *A Tool for Rating Text/Image/Audio Stimuli via 'LLMs'*

---

## Description

Evaluates stimuli using Large Language Models through the 'llmcoder' package. Supports multiple LLM providers: 'OpenAI', 'Anthropic', 'Ollama', 'LM Studio', 'DeepSeek', 'Groq', 'Mistral', and 'OpenAI-compatible' endpoints. Designed for research rating tasks. Stimuli can be plain text, local image/audio files, or image URLs.

**New in v1.3.1:** Probability-weighted scoring (`method = "weighted"`) following Brysbaert et al.

(2025). When enabled, the function retrieves log probabilities for all candidate rating tokens and computes a continuous score as $\Sigma(\text{rating} \times p(\text{rating}))$, which yields finer-grained estimates than the dominant (most-probable) token alone. The full probability distribution can be saved via `include_probs = TRUE`.

## Usage

```
generate_ratings(
  model = NULL,
  stim,
  prompt = "You are an expert rater. Limit your answer to numbers only.",
  question = "Please rate this:",
  scale = "1-7",
  temp = 0,
  n_iterations = 1,
  provider = c("openai", "anthropic", "ollama", "lmstudio", "deepseek", "groq",
    "mistral", "openrouter", "openai_compatible"),
  api_key = NULL,
  base_url = NULL,
  debug = FALSE,
  return_type = c("numeric", "text", "raw"),
  columns = NULL,
  method = c("dominant", "weighted"),
  top_logprobs = 5,
  include_probs = FALSE
)
```

## Arguments

| | |
|---|---|
| `model` | LLM model name (default varies by provider) |
| `stim` | Input stimulus: a plain text string, a local file path (image or audio), or an image URL starting with http(s)://. Supported local files: images (png, jpg, gif, webp, bmp), audio (mp3, wav, ogg, flac, m4a, aac). |
| `prompt` | System instruction for the LLM |
| `question` | Specific rating question for the LLM |
| `scale` | Rating scale range (default: '1-7'). Ignored when return_type = 'text' or 'raw'. |
| `temp` | Temperature parameter (default: 0) |
| `n_iterations` | Number of rating iterations (default: 1). Only used when `method = "dominant"` (temperature > 0 yields varied responses). |
| `provider` | LLM provider: 'openai', 'anthropic', 'ollama', 'lmstudio', 'deepseek', 'groq', 'mistral', 'openrouter', 'openai_compatible' |
| `api_key` | API key (not needed for local providers like 'ollama') |
| `base_url` | Custom base URL for OpenAI-compatible APIs |
| `debug` | Debug mode flag (default: FALSE) |
| `return_type` | Output type: 'numeric' (default, extract numbers only), 'text' (return full text response), 'raw' (return unprocessed response) |

| | |
|---|---|
| columns | Character vector of column names to include in the returned data frame. Available columns: 'stim', 'rating', 'iteration', 'scale', 'type', 'provider', 'model', 'probs_json'. Default is NULL (all columns returned). |
| method | Rating method: "dominant" (default) returns the most probable numeric token; "weighted" computes a probability-weighted average $\Sigma(\text{rating} \times p(\text{rating}))$ using log probabilities from the API. Only supported for provider = "openai" with text stimuli. See Brysbaert et al. (2025) doi:10.3758/s1342802402515z. |
| top_logprobs | Integer (0-20). Number of most likely tokens to return at each position (default: 5). Only used when method = "weighted". Higher values capture more token surface variants (e.g. "3", " 3", "3."). |
| include_probs | Logical. If TRUE, the returned data frame includes a probs_json column with the full probability distribution over rating values as a JSON string. Only used when method = "weighted". |

## Value

A data frame containing ratings and metadata. Columns depend on the columns argument. The 'rating' column type depends on return_type. When method = "weighted" and include_probs = TRUE, an additional probs_json column is included.

## Examples

```
## Not run:
# ------------------------------------------------------------------
# 1. TEXT STIMULUS - dominant rating (default, local LLM)
# ------------------------------------------------------------------
result <- generate_ratings(
  stim     = "kick the bucket",
  prompt   = "Rate how idiomatic this is (1-7):",
  scale    = "1-7",
  provider = "ollama"
)
result

# ------------------------------------------------------------------
# 2. TEXT STIMULUS - probability-weighted rating (OpenAI only)
# ------------------------------------------------------------------
# Follows Brysbaert et al. (2025): computes continuous score as
# sum(rating * p(rating)) using logprobs from the API.
result <- generate_ratings(
  stim         = "kick the bucket",
  prompt       = "You are a helpful assistant.",
  question     = "Rate familiarity (1-5):",
  scale        = "1-5",
  provider     = "openai",
  api_key      = Sys.getenv("OPENAI_API_KEY"),
  method       = "weighted",
  top_logprobs = 20,
  include_probs = TRUE
)
```

```
result$rating       # e.g. 3.72 (continuous, not just "3" or "4")
result$probs_json   # e.g. {"3":0.52,"4":0.43,"2":0.05}

# ------------------------------------------------------------------
# 3. TEXT STIMULUS — full text description
# ------------------------------------------------------------------
result <- generate_ratings(
  stim        = "spill the beans",
  prompt      = "You are an expert linguist.",
  question    = "Explain the meaning:",
  provider    = "openai",
  api_key     = Sys.getenv("OPENAI_API_KEY"),
  return_type = "text"
)

# ------------------------------------------------------------------
# 4. IMAGE STIMULUS — numeric rating
# ------------------------------------------------------------------
result <- generate_ratings(
  stim     = "/path/to/image.png",
  prompt   = "Rate visual complexity (1-5):",
  scale    = "1-5",
  provider = "openai",
  api_key  = Sys.getenv("OPENAI_API_KEY")
)

# ------------------------------------------------------------------
# 5. MULTIPLE ITERATIONS — reliability check
# ------------------------------------------------------------------
result <- generate_ratings(
  stim         = "once in a blue moon",
  prompt       = "Rate familiarity (1-7):",
  provider     = "openai",
  api_key      = Sys.getenv("OPENAI_API_KEY"),
  n_iterations = 3,
  columns      = c("stim", "rating", "iteration")
)

## End(Not run)
```

---

`generate_ratings_for_all` *Batch Rating Generator*

---

## Description

Process multiple stimuli in sequence using 'LLMs'. Supports text, image, and audio stimuli (see [generate_ratings()]).

## Usage

```
generate_ratings_for_all(model = NULL, stim_list, ...)
```

## Arguments

| | |
|---|---|
| model | LLM model name (default varies by provider) |
| stim_list | A character vector of stimuli to process. Each element can be a text string, a local file path (image/audio), or an image URL. |
| ... | Additional arguments passed to [generate_ratings()] |

## Examples

```
## Not run:
# ------------------------------------------------------------------
# 1. BATCH TEXT — rate multiple idioms for familiarity (dominant)
# ------------------------------------------------------------------
idioms <- c("kick the bucket", "spill the beans", "break a leg",
            "hit the nail on the head", "once in a blue moon")
results <- generate_ratings_for_all(
  stim_list = idioms,
  prompt    = "Rate how familiar this expression is (1-7):",
  scale     = "1-7",
  provider  = "ollama",
  columns   = c("stim", "rating")
)
results

# ------------------------------------------------------------------
# 2. BATCH RATING — probability-weighted scoring (v1.3.1)
# ------------------------------------------------------------------
# Follows Brysbaert et al. (2025): continuous scores from logprobs
results <- generate_ratings_for_all(
  stim_list    = idioms,
  prompt       = "You are a helpful assistant.",
  question     = "Rate familiarity (1-5):",
  scale        = "1-5",
  provider     = "openai",
  api_key      = Sys.getenv("OPENAI_API_KEY"),
  method       = "weighted",
  top_logprobs = 20,
  include_probs = TRUE,
  columns      = c("stim", "rating", "probs_json")
)
results

# ------------------------------------------------------------------
# 3. BATCH TEXT — get full descriptions
# ------------------------------------------------------------------
results <- generate_ratings_for_all(
  stim_list   = c("kick the bucket", "spill the beans"),
  prompt      = "You are an expert linguist.",
```

```
  question    = "Explain the meaning of this idiom in one sentence:",
  provider    = "openai",
  api_key     = Sys.getenv("OPENAI_API_KEY"),
  return_type = "text",
  columns     = c("stim", "rating")
)

# ------------------------------------------------------------------
# 4. BATCH IMAGE — mix of local files and URLs
# ------------------------------------------------------------------
stimuli <- c(
  "/path/to/image1.png",
  "/path/to/image2.jpg",
  "https://osf.io/download/example_img.png"
)
results <- generate_ratings_for_all(
  stim_list = stimuli,
  prompt    = "Rate the visual complexity (1 = simple, 5 = complex):",
  scale     = "1-5",
  provider  = "openai",
  api_key   = Sys.getenv("OPENAI_API_KEY"),
  columns   = c("stim", "rating", "type")
)

## End(Not run)
```

# Index