ellmer 0.4.0

Photo by Evan Jones

2025/11/18

ellmer

Hadley Wickham

We're very happy to announce the release of ellmer 0.4.0. ellmer makes it easy to chat with a large language model directly from R. It supports a wide variety of providers (including OpenAI, Anthropic, Azure, Google, Snowflake, Databricks and many more), makes it easy to extract structured data, and to give the LLM the ability to call R functions via tool calling.

You can install it from CRAN with:

```
install.packages("ellmer")
```

This blog post will cover the major changes in this release, including important lifecycle updates, new features for Claude (caching, file uploads, and web tools), improvements to OpenAI support (responses API and built-in tools), and a variety of enhancements to error handling, pricing tracking, and security.

You can see a full list of changes in the release notes.

```
library(ellmer)
```

Lifecycle

Contents

`batch_chat()` do a much better job of handling errors, and I'm confident that they're around to stay.

Reflecting Anthropic's recent rebranding of developer tools under the Claude name, `chat_claude()` is no longer deprecated and is an alias for `chat_anthropic()`. New `models_claude()` is now an alias for `models_anthropic()`.

The following deprecated functions/arguments/methods have been removed:

- `Chat$extract_data()` -> `chat$chat_structured()` (0.2.0)
- `Chat$extract_data_async()` -> `chat$chat_structured_async()` (0.2.0)
- `chat_anthropic(max_tokens)` -> `chat_anthropic(params)` (0.2.0)
- `chat_azure()` -> `chat_azure_openai()` (0.2.0)
- `chat_azure_openai(token)` (0.1.1)
- `chat_bedrock()` -> `chat_aws_bedrock()` (0.2.0)
- `chat_claude()` -> `chat_anthropic()` (0.2.0)
- `chat_cortex()` -> `chat_snowflake()` (0.2.0)
- `chat_gemini()` -> `chat_google_gemini()` (0.2.0)
- `chat_openai(seed)` -> `chat_openai(params)` (0.2.0)
- `create_tool_def(model)` -> `create_tool_def(chat)` (0.2.0)

`chat_claude()`

`chat_claude()` gains a new `cache` parameter to control caching. By default it is set to "5m". Claude's caching model is rather difficult to understand, but I'm reasonably confident that this will reduce your costs overall. `?chat_claude` goes into the details of why I think this will save you money.

With help from @dcomputing, ellmer has gained a suite of file management helpers such as `claude_file_upload()`,

You can now take advantage of Claude's built-in web search and web fetch with `claude_tool_web_search()` and `claude_tool_web_fetch()`. These empower Claude to perform web searches and read web pages on your behalf.

`chat_openai()` and `chat_openai_compatible()`

`chat_openai()` now uses OpenAI's more modern "responses API". This is their now-recommended API, and unlocks the ability to use the built-in tools, such as web search with `openai_tool_web_search()`. It also gains a `service_tier` argument which allows you to request slower/cheaper or faster/more expensive results.

If you want to talk to a model provider that is OpenAI API compatible (i.e. uses the older "chat completions" API), you'll need to use `chat_openai_compatible()`.

New features

- `parallel_chat()` and `batch_chat()` are much better at dealing with errors, and should now (by and large) succeed even if not all prompts succeeded or return badly formatted output. This does make the output from `parallel_chat()` a bit more complex, since it can now be a mix of `Chat` objects, error objects, and `NULL`, but we think the trade-off is worth it.

- `batch_chat()` and friends have a revised hashing mechanism which is used to ensure that you don't accidentally use saved results with the wrong inputs. The mechanism now only hashes the provider `name`, `model`, and `base_url`. This should provide some protection from accidentally reusing the same `.json` file with different providers, while still allowing you to use the same batch file across ellmer versions. There's also a new `ignore_hash`

changed.

- There were a bunch of smaller improvements to pricing: the package now uses the latest pricing data, `batch_chat()` only records costs on retrieval, `Chat$get_tokens()` includes cost information, and the print method does a better job of matching underlying data.

- `params()` gains new `reasoning_effort` and `reasoning_tokens` so you can control the amount of effort a reasoning model spends on thinking. Initial support is provided for `chat_claude()`, `chat_google_gemini()`, and `chat_openai()`.

- `chat_*()` functions now use a `credentials` function instead of an `api_key` value. This means that API keys are never stored in the chat object (which might be saved to disk), but are instead retrieved on demand as needed. You generally shouldn't need to use the `credentials` argument directly yourself, but when you do, you should use it to dynamically retrieve the API key from some other source (i.e. never inline a secret directly into a function call).

- `tool()`s can now return image or PDF content types, with `content_image_file()` or `content_pdf()`.

- You can use the new `schema_df()` to describe the schema of a data frame to an LLM. It's designed to give a high-quality summary without spending too many tokens.

Acknowledgements

A big thanks to everyone who contributed to this release! @abiyug, @AdaemmerP, @AlmogAngel, @app2let, @benhmin, @bensoltoff, @benzipperer, @bianchenhao, @bshor, @CChen89, @cherylisabella, @cpsievert, @dcomputing, @durraniu, @fh-slangerman, @flaviaerius, @foton263, @gadenbuie, @gary-mu, @Green-State-Data, @hadley, @howardbaik, @jeroenjanssens, @jharvey-records, @joranE, @kbenoit, @LukasWallrich, @m20m22, @maciekbanas, @mattwarkentin, @parmsam, @parmsam-pfizer, @promothesh, @rempsyc, @roldanalex,

The tidyverse is proudly supported by