



FUTURE OF
COMPUTING
INSTITUTE

LLMs and R: How Posit is Making it Easy!

Wednesday, 12 Nov 2025

RPIrates: The RPI R Users Group
The Rensselaer Future of Computing Institute
Rensselaer Polytechnic Institute

Recalling a Simpler Time...

RPIrates Feb 2023



<https://idea.rpi.edu/media/talks/2023/rpirates-fun-openai-gptstudio-and-r>



ChatBS and ChatBS-NextGen

ChatBS: A Context-aware LLM Exploratory Sandbox



Rensselaer

A Context-aware LLM Exploratory Sandbox

<https://inciteprojects.idea.rpi.edu/chatbs/app/chatbs/>



CHATBS:
An LLM Exploratory Sandbox

System Prompt

You are a helpful assistant

User Prompt

What is the oldest technical university in the United States?

Number of re-submissions

1

LLM model (multi-select to compare models)

Meta-Llama-3.1-8B-Instruct

☐ Ask LLM to 'explain step-by-step' NOTE: Lengthens response time!

ChatBS, the Context-aware LLM Exploratory Sandbox uses the OpenAI Completion API service (GPT and Llama family models) to answer questions. Each sentence in a ChatBS result is automatically linked to a Google query to facilitate fact-checking. If requested, ChatBS can then use the OpenAI API to construct an entity/relation graph of these results in the form ['entity1', 'relationship', 'entity2']. ChatBS then uses entity linking to look up both the entities and relationships against Wikidata entities and properties, constructing a JSON-LD graph as it proceeds.

For each re-submission of the user's prompt we display the LLM response and the resulting RDF (if requested)

LLM Results:

Based on: Meta-Llama-3.1-8B-Instruct

Click on any sentence in results to 'fact-check' with Google (opens new window)

Result 1: [Meta-Llama-3.1-8B-Instruct] The oldest technical university in the United States is Rensselaer Polytechnic Institute (RPI), which was founded in 1824 in Troy, New York. It was established by Stephen Van Rensselaer III, a member of the Van Rensselaer family, who provided a founding endowment of \$40,000. RPI was originally called the Rensselaer School and was established to provide training in the subjects of art and science. Over time, the institution evolved to become a full-fledged technical university, offering a wide range of programs in engineering, science, and technology. Today, RPI is a highly respected institution that is known for its rigorous academic programs, cutting-edge research, and innovative spirit.

Download complete results as JSON file

Summary of relationships (RDF) found for these results:

Click on a row for more details about that row's subject entity. Objects highlighted in gray are nodes (blank/unlabelled nodes) in the graph.

2024

ChatBS-NexGen: Automating Workflows for LLM Evaluation over Knowledge Graphs



ChatBS-NexGen

Experiment Run Setup Pipeline Overview

Experiment Run

This page allows users to run tests in Answer Generation or Fact Verification mode across multiple LLMs, datasets, and retries.

Pipeline

Mode: ☐ Answer Generation ☒ Fact Extraction

Upload Pipeline Configuration File

Browse: pipeline_recipe_fact_extraction_final.json Upload complete

Dataset

Upload Dataset File

Browse: patient_dataset.csv Upload complete

Settings

Number of Retries

1

LLMs

Meta-Llama-3.1-8B-Instruct

Validation are successfully passed. Experiment Started ...

Run Experiment

Experiment Results:

Download experiment results as JSON file

Show: 5 entries

	llm	query_id	try	user_prompt	system_prompt	generated_answerextracted_info	ingredient	GI
1	Meta-Llama-3.1-8B-Instruct	1	1	I need suggest...	You are a he...	Based on yo...	carbohydrate...	fruit, fat, avoca...
2	Meta-Llama-3.1-8B-Instruct	2	1	I need suggest...	You are a he...	Based on yo...	vegetable, fru...	fruit, grain, po...
3	Meta-Llama-3.1-8B-Instruct	3	1	I need suggest...	You are a he...	Based on yo...	vegetable, fru...	fruit, grain, so...

Showing 1 to 3 of 3 entries

Previous 1 Next

Selected Row Details:

Field	Value
llm	Meta-Llama-3.1-8B-Instruct
query_id	3
try	1
user_prompt	I need suggestions for light and energy-blasting foods to eat before my gym workout.
system_prompt	You are a health care assistant and you will respond to the user. 35 years old. Male, and None, who is a beneficiary of the health care program. Tailor dietary guidelines by considering recent clinical metrics. Height: 181.4, Weight: 104.2, BMI: 24.9, HbA1c: 5.6, Blood Pressure: 120/80 mmHg, HDL: 40.7, LDL: 117.1, Fasting Glucose: 111, Waist Circumference: 102. Account for individual food preferences. Vegan and allergies. Tree nuts. Include strategies for weight adjustment. 81.5550404.
generated_answer	Based on your clinical metrics, dietary guidelines, and individual food preferences, I recommend the following tailored plan: "Objective:" Maintain a healthy weight, improve lipid profiles, and reduce cardiovascular risk. "Caloric Intake:" To promote weight loss, aim for a daily caloric deficit of 500-750 calories. Based on your Basal Metabolic Rate (BMR), a sustainable daily caloric intake would be 2500-2800 calories. "Macronutrient Breakdown:" Carbohydrates: 55-60% of total daily calories (150-170g). Protein: 15-20% of total daily calories (75-90g). Fat: 20-25% of total daily calories (40-50g). "Food Choices:" 1. "Vegetables:" Aim for at least 5 servings (2.5 cups) of a variety of colorful vegetables daily. Include leafy greens, bell peppers, carrots, and broccoli. 2. "Fruits:" Eat at least 3 servings (1.5 cups) of different types of fruits daily. Choose fruits rich in fiber and antioxidants like berries, citrus fruits, and apples. 3. "Protein Sources:" Legumes (lentils, chickpeas, black beans, and kidney beans) Whole grains (quinoa, brown rice, and whole-wheat bread) Nuts and seeds (except tree nuts due to the allergy) Tofu and other plant-based protein sources 4. "Healthy Fats:" Avocado Olive oil Fatty fish (salmon, tuna, and sardines) 5. "Whole Grains:" Brown rice Quinoa Whole-wheat bread Whole-grain pasta "Strategies for Weight Adjustment:" 1. Drink plenty of water throughout the day (at least 8 cups). 2. Incorporate physical activity for weight loss should begin slowly with a minimum of 150 minutes/week of moderate-intensity aerobic activity or 75 minutes/week of vigorous-intensity aerobic activity, or an equivalent combination of both, preferably spread throughout the week. 3. Incorporate strength-training exercises 2-3 times a week to build muscle mass and boost metabolism. 4. Incorporate healthy snacking ideas into your daily routine to curb hunger and support weight loss, suggested snacks: fresh fruits, veggie sticks with hummus. Energy balls made from oats, nuts, and dried fruits. 5. Prioritize stress management techniques, such as meditation, deep breathing, or yoga, to help regulate hunger.

2025

Posit Generative AI Solutions

Posit provides a range of tools and LLM-related packages for R and Python

Access LLMs



Build Shiny apps
that chat with LLMs



Evaluate LLMs



Posit Generative AI Solutions

Posit provides a range of tools and LLM-related packages for R and Python

Enhance LLM workflows



IDE Integrations



Automate and improve data science tasks with GenAI





ellmer

ellmer makes it easy to use large language models (LLM) from R. It supports a wide variety of LLM providers and implements a rich set of features including streaming outputs, tool/function calling, structured data extraction, and more.

ellmer is one of a number of LLM-related packages created by Posit:

- Looking for something similar in python? Check out [chatlas](#)!
- Want to evaluate your LLMs? Try [vitals](#).
- Need RAG? Take a look at [ragnar](#).
- Want to make a beautiful LLM powered chatbot? Consider [shinychat](#).
- Working with MCP? Check out [mcp tools](#).



ellmer

Providers

ellmer supports a wide variety of model providers:

- Anthropic's Claude: `chat_anthropic()`.
- AWS Bedrock: `chat_aws_bedrock()`.
- Azure OpenAI: `chat_azure_openai()`.
- Cloudflare: `chat_cloudflare()`.
- Databricks: `chat_databricks()`.
- DeepSeek: `chat_deepseek()`.
- GitHub model marketplace: `chat_github()`.
- Google Gemini/Vertex AI: `chat_google_gemini()`, `chat_google_vertex()`.
- Groq: `chat_groq()`.
- Hugging Face: `chat_huggingface()`.
- Mistral: `chat_mistral()`.
- Ollama: `chat_ollama()`.
- OpenAI: `chat_openai()`.
- OpenRouter: `chat_openrouter()`.
- perplexity.ai: `chat_perplexity()`.
- Snowflake Cortex: `chat_snowflake()` and `chat_cortex_analyst()`.
- VLLM: `chat_vllm()`.



ellmer

Provider/model choice

If you're using ellmer inside an organisation, you may have internal policies that limit you to models from big cloud providers, e.g. `chat_azure_openai()`, `chat_aws_bedrock()`, `chat_databricks()`, or `chat_snowflake()`.

If you're using ellmer for your own exploration, you'll have a lot more freedom, so we have a few recommendations to help you get started:

- `chat_openai()` or `chat_anthropic()` are good places to start. `chat_openai()` defaults to **GPT-4.1**, but you can use `model = "gpt-4-1-nano"` for a cheaper, faster model, or `model = "o3"` for more complex reasoning. `chat_anthropic()` is also good; it defaults to **Claude 4.0 Sonnet**, which we have found to be particularly good at writing R code.
- `chat_google_gemini()` is a strong model with generous free tier (with the downside that your data is used to improve the model), making it a great place to start if you don't want to spend any money.
- `chat_ollama()`, which uses [Ollama](#), allows you to run models on your own computer. While the biggest models you can run locally aren't as good as the state of the art hosted models, they don't share your data and are effectively free.



vitals

vitals is a framework for large language model evaluation in R. It's specifically aimed at [ellmer](#) users who want to measure the effectiveness of their LLM products like [custom chat apps](#) and [querychat](#) apps. You can use it to:

- Measure whether changes in your prompts or additions of new tools improve performance in your LLM product
- Compare how different models affect performance, cost, and/or latency of your LLM product
- Surface problematic behaviors in your LLM product

The package is an R port of the widely adopted Python framework [Inspect](#). While the package doesn't integrate with Inspect directly, it allows users to interface with the [Inspect log viewer](#) and provides an on-ramp to transition to Inspect if need be by writing evaluation logs to the same file format.



ragnar

`ragnar` is an R package that helps implement Retrieval-Augmented Generation (RAG) workflows. It focuses on providing a complete solution with sensible defaults, while still giving the knowledgeable user precise control over each step. We don't believe that you can fully automate the creation of a good RAG system, so it's important that `ragnar` is not a black box. `ragnar` is designed to be transparent. You can easily inspect outputs at intermediate steps to understand what's happening.



ragnar

Key Steps

1. Document Processing

`ragnar` works with a wide variety of document types, using [MarkItDown](#) to convert content to Markdown.

Key functions:

- `read_as_markdown()`: Convert a file or URL to markdown
- `ragnar_find_links()`: Find all links in a webpage

2. Text Chunking

Next we divide each document into chunks. Ragnar defaults to a strategy that preserves some of the semantics of the document, but provides plenty of opportunities to tweak the approach.

Key functions:

- `markdown_chunk()`: Full-featured chunker that identifies semantic boundaries and intelligently chunks text.



ragnar

3. Context Augmentation (Optional)

RAG applications benefit from augmenting text chunks with additional context, such as document headings and subheadings. `ragnar` makes it easy to keep track of headings and subheadings as part of chunking.

`markdown_chunk()` automatically associates each chunk with the headings that are in scope for that chunk.

4. Embedding

`ragnar` can help compute embeddings for each chunk. The goal is for `ragnar` to provide access to embeddings from popular LLM providers.

Key functions:

- `embed_ollama()`
- `embed_openai()`
- `embed_bedrock()`
- `embed_databricks()`
- `embed_google_vertex()`

Note that calling the embedding function directly is typically not necessary. Instead, the embedding function is specified when a store is first created, and then automatically called when needed by `ragnar_retrieve()` and `ragnar_store_insert()`.



ragnar

5. Storage

Processed data is stored in a format optimized for efficient searching, using `duckdb` by default. The API is designed to be extensible, allowing additional packages to implement support for different storage providers.

Key functions:

- `ragnar_store_create()`
- `ragnar_store_connect()`
- `ragnar_store_insert()`

6. Retrieval

Given a prompt, retrieve related chunks based on embedding distance or bm25 text search.

Key functions:

- `ragnar_retrieve()`: high-level function that performs both `vss` and `bm25` search and de-overlaps retrieved results.
- `ragnar_retrieve_vss()`: Retrieve using [vss DuckDB extension](#)
- `ragnar_retrieve_bm25()`: Retrieve using [full-text search DuckDB extension](#)
- `chunks_deoverlap()`: Consolidates retrieved chunks that overlap.

7. Chat Augmentation

`ragnar` can equip an `ellmer::Chat` object with a retrieve tool that enables an LLM to retrieve content from a store on-demand.

Also RAGNAR

- The Relay Races: <https://runragnar.com/>

- The Legend

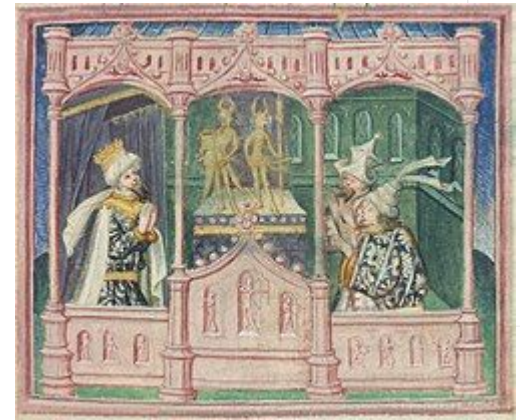
Ragnar Lodbrok

[Article](#) [Talk](#)

From Wikipedia, the free encyclopedia

Ragnar Lodbrok (*Old Norse*: *Ragnarr loðbrók*, *lit.* 'Ragnar hairy-breeches'),^[a] according to *legends*,^[2] was a *Viking* hero and a *Swedish* and *Danish* king.^[3]

He is known from *Old Norse poetry* of the *Viking Age*, Icelandic *sagas*, and near-contemporary chronicles. According to traditional literature, Ragnar distinguished himself by conducting many *raids* against the *British Isles* and the *Carolingian Empire* during the 9th century. He also appears in *Norse legends*, and according to the *legendary sagas* *Tale of Ragnar's Sons* and a *Saga about Certain Ancient Kings*,^[4] Ragnar Lodbrok's father has been given as the legendary king of the *Swedes*, *Sigurd Ring*.^{[5][6]}



shinychat



shinychat provides a [Shiny](#) toolkit for building generative AI applications like chatbots and [streaming content](#). It's designed to work alongside the [ellmer](#) package, which handles response generation.

Installation

You can install shinychat from CRAN with:

```
install.packages("shinychat")
```

Or, install the development version of shinychat from [GitHub](#) with:

IDE Integrations: chattr

chattr



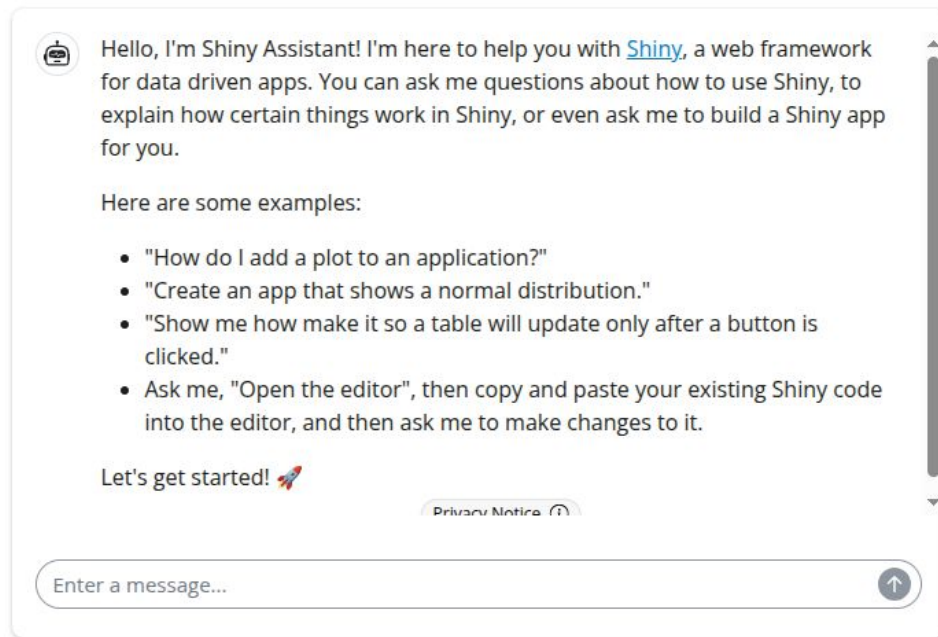
- [Intro](#)
- [Install](#)
- [Using](#)
 - [Available models](#)
 - [The App](#)
 - [Additional ways to interact](#)
- [How it works](#)
- [Keyboard Shortcut](#)
 - [How to setup the keyboard shortcut](#)

Intro

`chattr` is an interface to LLMs (Large Language Models). It enables interaction with the model directly from RStudio and Positron. `chattr` allows you to submit a prompt to the LLM from your script, or by using the provided Shiny Gadget.

This package's main goal is to aid in exploratory data analysis (EDA) tasks. The additional information appended to your request, provides a sort of "guard rails", so that the packages and techniques we usually recommend as best practice, are used in the model's responses.

IDE Integrations: Shiny Assistant



Access LLMs in R and Python



ellmer

The [ellmer](#) package simplifies LLM integration in R. Ellmer provides a consistent interface to multiple LLM providers, with features like streaming, tool calling, and structured data extraction.

- [Documentation](#)
- [Release blog post](#)

chatlas

The [chatlas](#) package simplifies LLM integration in Python. Chatlas offers a unified interface across LLM providers for tasks like streaming chats, tool calling, and structured output.

- [Documentation](#)
- [Release blog post](#)

ragnar

The [ragnar](#) package enhances LLM performance with Retrieval-Augmented Generation (RAG) in R, which allows you to retrieve relevant external data to inform responses.

- [Documentation](#)
- [Release blog post](#)

Enhance your LLM workflow



btw

The btw package helps you describe your computational environment to LLMs.

- [Documentation](#)
- [Technical introduction](#)

vitals

The vitals package provides a framework for large language model evaluation in R.

- [Documentation](#)
- [Technical introduction](#)

mcptools

The mcptools package implements a [Model Context Protocol](#) (MCP) server for your R sessions.

- [Documentation](#)
- [Technical introduction](#)

Build Shiny apps that chat with LLMs



Shinychat

Shinychat provides components integrate interactive chat interfaces into Shiny applications with a dedicated UI component.

- [R documentation](#)
- [Python documentation](#)

querychat

querychat is a drop-in component for Shiny that allows users to query a data frame using natural language.

- [Documentation](#)

Automate and improve data science tasks with GenAI



chores

The [chores](#) package automates repetitive coding tasks with LLM-powered assistants. Use customizable shortcuts to trigger code rewrites and enhancements directly in your R environment.

- [Documentation](#)
- [Technical Introduction](#)

gander

The [gander](#) package enhances data science workflows in RStudio and Positron with intelligent, in-line LLM chats. Gander integrates Ellmer chats into your projects, providing context-aware responses and streaming results directly into your documents.

- [Documentation](#)
- [Technical Introduction](#)

mall

The [mall](#) package applies LLM predictions to data frames efficiently. Process rows with pre-defined prompts for batch analysis in both R and Python.

- [Documentation](#)
- [Technical Introduction](#)

lang

The [lang](#) package translates R function help documentation on-the-fly to other languages such as Spanish or French. Lang overrides help functions to provide instant translations within RStudio and Positron.

- [Documentation](#)

IDE Integrations

Positron Assistant

Experimental feature in Positron 2025.06.0

Positron Assistant transforms [Positron](#) into an AI-powered data science workspace, deeply integrated with your tools and workflows while leveraging enterprise-grade LLM providers.

- [User Guide](#)
- [Announcement post](#)

Databot

Databot is an interactive AI agent designed specifically for exploratory data analysis. Databot actively drives your data exploration through a tight WEAR (write, execute, analyze, regroup) loop.

- [Announcement post](#)

Shiny Assistant

[Shiny Assistant](#) accelerates Shiny development with an AI assistant. Get help with Shiny questions, create applications from scratch, or modify existing code in R and Python.

- [Shiny Assistant](#)
- [Release blog post](#)

GitHub Copilot

[GitHub Copilot](#) enhances coding productivity with AI-powered code suggestions directly within RStudio.

- [Documentation](#)
- [Release blog post](#)

[chattr](#)

The [chattr](#) package allows you to interact with LLMs directly from RStudio. Submit prompts from scripts or use the Shiny gadget for interactive conversations.

- [Documentation](#)
- [Technical introduction](#)

Enable effective observability

otel

Use the otel package as a dependency if you want to instrument your R package or project for OpenTelemetry.

otelsdk

Use the otelsdk package to produce OpenTelemetry output from an R package or project that was instrumented with the otel package.

AI demonstrations, use cases, and best practices

- [Harnessing LLMs for Data Analysis | Led by Joe Cheng, CTO at Posit](#)
- [Easy tool calls with ellmer and chatlas](#)
- [Natural language data science with RStudio and Databricks](#)
- [How to use natural language data science with RStudio and Amazon SageMaker](#)
- [Announcing secure AI-Assisted data science in R with Posit and Snowflake Cortex](#)
- [AI tool built with Posit decreases unexpected deaths of hospitalized patients by 26%](#)
- [Generate data with an LLM and ellmer](#)
- [Text Summarization, Translation, and Classification using LLMs](#)
- [The potential for AI-powered Shiny app prototyping with Shiny Assistant](#)
- [Running AI/LLM Hackathons at Posit: What We've Learned](#)
- [Trail Running Meets Data Science: Adventures with LLMs and Race Stats](#)
- [Setting up local LLMs for R and Python](#)
- [What LLMs Actually Do \(and What They Don't\)](#)