




# chatAI4R: Interactive Artificial Intelligence toolkit for Data Science in R

Satoshi Kume <sup>1</sup>✉

<sup>1</sup> Bio"Pack"athon, Osaka, Japan ✉ Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Chris Vernon](#)  

## Reviewers:

- [@tzuV](#)
- [@khoa-yelo](#)

Submitted: 02 July 2025

Published: unpublished

## License

Authors of papers retain copyright  
and release the work under a  
Creative Commons Attribution 4.0  
International License ([CC BY 4.0](#))

## Summary

Large Language Models (LLMs) have revolutionized natural language processing (NLP), data mining, and program coding. The chatAI4R package provides a comprehensive toolkit for seamlessly integrating LLMs within R environments. Beyond basic text generation and conversation capabilities, it supports text embeddings and delivers sophisticated LLM assistance through simple function calls, significantly extending R-based data analysis and knowledge discovery processes. Unlike existing R packages, the chatAI4R package offers unique R package development support features. Rather than functioning as a multi-functional API wrapper, it provides comprehensive development automation and AI-assisted data mining capabilities. The package combines command-line and graphical operations, offering flexibility for users across all skill levels. Available on both GitHub and the Comprehensive R Archive Network (CRAN), chatAI4R ensures stability, reliability, and broad community accessibility.

Originally a development aid for R packages since 2023, chatAI4R has evolved into a data-analysis companion by adding interpretation, knowledge extraction, and multi-LLM capabilities. It serves R package developers (automating Roxygen2 docs, function generation, code quality) and data analysts (statistical interpretation, literature processing, insight extraction) with a four-layer architecture that goes beyond multi-functional API wrappers.

## State of the Field

Since GPT-4's release ([OpenAI, 2024](#)), LLMs have rapidly evolved, transforming NLP, data analysis, and programming approaches. While chat-based interfaces offer intuitive experiences, they are insufficient for complex analytical tasks requiring multi-step processing and statistical integration. Current AI agents still suffer from response time limitations in their speculative processing, which makes them unsuitable for iterative workflows. Therefore, direct programmatic access through R becomes essential, leveraging its rich statistical ecosystem and creating a need for specialized R packages that provide efficient LLM integration for data science applications.

The R ecosystem now includes several LLM-focused packages with distinct approaches. For comprehensive LLM integration, ellmer ([Wickham et al., 2025](#)) provides wide provider support with advanced features including streaming outputs, tool calling, and structured data extraction. Basic API access is offered by packages like openai (comprehensive but now archived) ([Rudnyskiy, 2024](#)) and gprr ([Gu, 2024](#)), which provides a simple interface through its `get_response()` function for straightforward ChatGPT interactions.

For local LLM deployment, both ollamar ([Lin & Safi, 2025](#)) and rollama ([Gruber & Weber, 2024](#)) facilitate integration with Ollama, an open-source framework for running local LLMs, enabling private and reproducible model execution focused on text annotation and document embedding capabilities.

41 Development-focused packages include chatgpt (Rodriguez, 2023) and gptstudio (Nivard et al.,  
42 2024), both providing RStudio addins for coding assistance. While chatgpt focuses specifically  
43 on OpenAI integration with features like code commenting, auto-completion, and Roxygen2  
44 documentation generation, gptstudio (Nivard et al., 2024) offers broader provider support  
45 through a unified interface.

46 While these existing tools provide valuable functionality, they primarily serve as API wrappers  
47 or development assistants, leaving significant gaps in comprehensive R-specific package  
48 development support and integrated data analysis workflows. These limitations create an  
49 opportunity for a more comprehensive solution that chatAI4R aims to address.

## 50 Statement of need

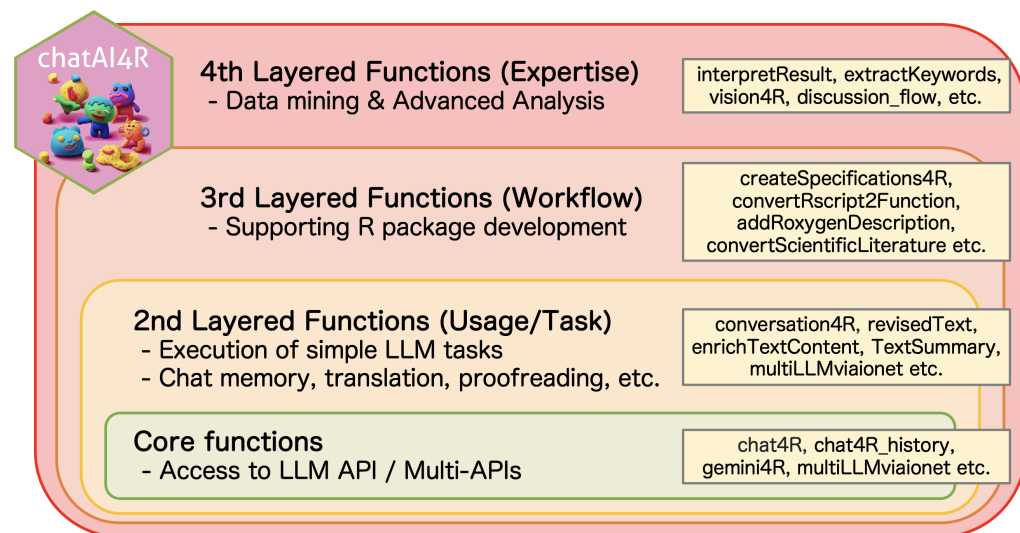
51 While existing R packages provide basic LLM functionality, critical gaps remain in comprehensive  
52 R-specific package development support and integrated data analysis workflows. Current tools  
53 serve as general-purpose API wrappers without addressing complex analytical needs.

54 chatAI4R addresses these limitations through its unique multi-layered conceptual architecture,  
55 providing a comprehensive ecosystem for LLM integration specifically designed for R users. The  
56 package supports nine LLM API platforms through unified interfaces, including OpenAI GPT  
57 models and Google Gemini, and provides innovative access to 18 models simultaneously via  
58 io.net's Intelligence API (<https://io.net/>, as of 14-JAN-2026), a cloud platform that provides  
59 distributed GPU computing resources, enabling access to state-of-the-art open-source models  
60 such as gpt-oss-120b, DeepSeek-R1, Qwen3, and Llama-4.

61 The package's core innovation lies in R-specific package development automation. Beyond basic  
62 text generation, chatAI4R offers automated R code generation through `createRfunction()`,  
63 intelligent comment addition via `addCommentCode()`, automatic Roxygen2 documentation  
64 (R's standard documentation format) with `addRoxygenDescription()`, and comprehensive  
65 package architecture planning through `designPackage()`, which assists in proposing the overall  
66 design and architecture of an R package. These capabilities transform LLMs into powerful  
67 development assistants tailored for R programming workflows.

68 A distinctive feature is the multi-agent discussion system (`discussion_flow_v1()`), where  
69 three specialized AI agents—the Beginner Bot, the Expert Bot, and the Peer Reviewer  
70 Bot—collaborate through Socratic dialogue (an iterative question-and-answer methodology).  
71 This approach enables iterative solution refinement with human intervention at critical decision  
72 points, addressing the single-shot interaction limitations present in existing tools.

73 The chatAI4R package excels in data analysis interpretation through the `interpretResult()`  
74 function, providing specialized interpretation for multiple analysis types including PCA and  
75 regression. This feature bridges the gap between statistical output and scientific interpretation,  
76 a capability that is absent in current R-LLM packages.



**Figure 1:** Figure 1: Four-layered conceptual framework of the chatAI4R package, showing the hierarchical structure from core functions to specialized applications

77 The package maintains production-level reliability through CRAN distribution, ensuring multi-  
78 platform compatibility and rigorous testing. Its open-source nature under the Artistic License  
79 2.0 promotes community contribution while maintaining professional development practices.

## 80 Design

81 The chatAI4R package implements a four-layered application architectural design that provides  
82 progressive functionality. This modular approach ensures accessibility for all users.

83 The Core Functions (Layer 1) establish unified API interfaces for multiple LLM platforms.  
84 The chat4R(), chat4R\_history(), gemini4R(), multiLLMviaionet(), and related functions  
85 provide standardized access patterns while handling authentication, rate limiting, and error  
86 management.

87 The Advanced Functions (2nd Layered Functions, Usage/Task) enable execution of simple  
88 LLM tasks through intelligent prompt engineering. Functions such as conversation4R(),  
89 revisedText(), enrichTextContent(), TextSummary(), and multiLLMviaionet() provide  
90 chat memory management, translation, proofreading, and text processing capabilities.

91 The Workflow Functions (3rd Layered Functions, Workflow) focus on supporting R  
92 package development. The createSpecifications4R(), convertRscript2Function(),  
93 addRoxygenDescription(), and convertScientificLiterature() functions provide  
94 comprehensive development automation, transforming LLMs into powerful assistants tailored  
95 for R programming workflows.

96 The Integration Functions (4th Layered Functions, Expertise) provide data mining and advanced  
97 analysis capabilities, while also providing connectivity with R ecosystem tools and development  
98 workflows. The interpretResult() function employs specialized templates for 13 analysis  
99 types, automatically generating domain-appropriate interpretations. The extractKeywords()  
100 extracts keywords from text, vision4R() enables image analysis via Vision API, and the multi-  
101 agent discussion system (discussion\_flow\_v1(), discussion\_flow\_v2()) employs role-based  
102 prompt engineering where each agent maintains distinct personas, enabling iterative solution  
103 refinement.

104 The package supports integration with six API platforms: OpenAI, Google Gemini, DeepL

translation, Replicate, Dify, and IO.net Intelligence API. This comprehensive API connectivity ensures flexibility in model selection and deployment scenarios. The package also facilitates the deployment of R-based web APIs by combining chatAI4R with the plumber package, allowing users to implement LLM-powered backend processing embedded in external platforms. Additionally, it supports GUI-based interactions, ensuring accessibility for users who prefer graphical interfaces over command-line operations.

The package employs defensive programming with comprehensive error handling, input validation, and graceful degradation when API services are unavailable. Package reliability is ensured through automated testing. Furthermore, to address the challenge of frequent API changes across these platforms, chatAI4R employs a sustainable maintenance strategy combining user contributions with high-frequency, LLM-assisted autonomous code updates, ensuring long-term resilience.

## Usage

The chatAI4R package provides multiple interaction modes to accommodate different user preferences and workflows. Users can access LLM capabilities through simple function calls or interactive interfaces.

Basic text generation and conversation capabilities are accessible through the core functions:

```
# Conversation with OpenAI GPT models
result <- chat4R(content = "Explain principal component analysis", Model = "gpt-5-nano")
```

```
# Multi-model comparison
result <- multiLLMviaionet(prompt = "Optimize this statistical analysis", max_models = 3)
```

For data analysis interpretation, the package provides specialized interpretations by specifying analysis types:

```
# Statistical result interpretation
pca_result <- prcomp(mtcars)
interpretation <- interpretResult(analysis_type = "PCA", result_text = summary(pca_result))
```

The multi-agent discussion system enables collaborative problem-solving through structured dialogue:

```
# Multi-agent collaboration (v1: basic)
discussion_flow_v1(
  issue = "Optimize machine learning pipeline",
  Domain = "data science"
)

# Multi-agent collaboration (v2: with extended settings)
discussion_flow_v2(
  issue = "Optimize machine learning pipeline",
  Domain = "data science",
  Sentence_difficulty = 2,
  R_expert_setting = TRUE,
  rep_x = 3
)
```

Development automation features streamline R package creation and maintenance:

```
# Function creation from clipboard content
# (Prerequisite: Copy "Function to calculate BMI" to clipboard)
createRfunction(Model = "gpt-5", SelectedCode = FALSE)
```

```
# Add Roxygen2 documentation
# (Prerequisite: Select the function definition below)
# my_add <- function(x, y) { x + y }
addRoxygenDescription(Model = "gpt-5-nano", SelectedCode = TRUE)

# Generate and Improve R Functions
autocreateFunction4R(
  Func_description = "Calculate Body Mass Index (BMI) given height and weight",
  Model = "gpt-5"
)
```

## References

- 127
- 128 Gruber, J. B., & Weber, M. (2024). *Rollama: An r package for using generative large language*  
129 *models through ollama*. <https://doi.org/10.48550/arXiv.2404.07654>
- 130 Gu, W. (2024). *Gptr: A convenient r interface with the OpenAI 'ChatGPT' API*. <https://doi.org/10.32614/cran.package.gptr>  
131
- 132 Lin, H., & Safi, T. (2025). *Ollamar: An r package for running large language models*. *Journal*  
133 *of Open Source Software*, 10(105), 7211. <https://doi.org/10.21105/joss.07211>
- 134 Nivard, M., Wade, J., & Calderon, S. (2024). *Gptstudio: Use large language models directly*  
135 *in your development environment*. <https://doi.org/10.32614/CRAN.package.gptstudio>
- 136 OpenAI, O. (2024). *GPT-4 technical report*. <https://doi.org/10.48550/arXiv.2303.08774>
- 137 Rodriguez, J. C. (2023). *Chatgpt: Interface to 'ChatGPT' from r*. [https://doi.org/10.32614/](https://doi.org/10.32614/CRAN.package.chatgpt)  
138 [CRAN.package.chatgpt](https://doi.org/10.32614/CRAN.package.chatgpt)
- 139 Rudnytskyi, I. (2024). *Openai: R wrapper for OpenAI API*. [https://doi.org/10.32614/cran.](https://doi.org/10.32614/cran.package.openai)  
140 [package.openai](https://doi.org/10.32614/cran.package.openai)
- 141 Wickham, H., Cheng, J., Jacobs, A., & Aden-Buie, G. (2025). *Ellmer: Chat with large*  
142 *language models*. <https://doi.org/10.32614/CRAN.package.ellmer>