Package: chatAI4R (via r-universe)

September 6, 2024

Type Package

Title Chat-Based Interactive Artificial Intelligence for R

Version 0.3.2

Date 2024-05-10

Maintainer Satoshi Kume <satoshi.kume.1984@gmail.com>

Description The Large Language Model (LLM) represents a groundbreaking advancement in data science and programming, and also allows us to extend the world of R. A seamless interface for integrating the 'OpenAI' Web APIs into R is provided in this package. This package leverages LLM-based AI techniques, enabling efficient knowledge discovery and data analysis (see 'OpenAI' Web APIs details <https://openai.com/blog/openai-api>). The previous functions such as seamless translation and image generation have been moved to other packages 'deepRstudio' and 'stableDiffusion4R'.

Depends R (>= 4.2.0)

Imports httr, jsonlite, assertthat, clipr, crayon, rstudioapi, future, igraph, deepRstudio, pdftools, xml2, rvest

Suggests testthat, knitr

License Artistic-2.0

URL https://kumes.github.io/chatAI4R/, https://github.com/kumeS/chatAI4R

BugReports https://github.com/kumeS/chatAI4R/issues

RoxygenNote 7.3.1

Encoding UTF-8

Repository https://kumes.r-universe.dev

RemoteUrl https://github.com/kumes/chatai4r

RemoteRef HEAD

RemoteSha da56b6c54001f0ae0b92750114c19fd2a606f93f

Contents

addCommentCode *Add Comments to R Code*

Description

This function adds comments to R code without modifying the input R code. It can either take the selected code from RStudio or read from the clipboard.

Usage

```
addCommentCode(Model = "gpt-4-0613", language = "English", SelectedCode = TRUE)
```

Arguments

Model	A character string specifying the GPT model to be used. Default is "gpt-4-0613".
language	A character string specifying the language for the comments. Default is "English".
SelectedCode	A logical value indicating whether to use the selected code in RStudio. Default is TRUE.

Details

Add Comments to R Code

Value

A message indicating completion if 'SelectedCode' is TRUE, otherwise the commented code is copied to the clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
addCommentCode(Model = "gpt-4-0613", language = "English", SelectedCode = TRUE)

## End(Not run)
```

addRoxygenDescription *Add Roxygen Description to R Function*

Description

This function adds a Roxygen description to an R function using the GPT-4 model. It can either take the selected code from RStudio or read from the clipboard.

Usage

```
addRoxygenDescription(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE
)
```

Arguments

Model	A character string specifying the GPT model to be used. Default is "gpt-4-0613".
SelectedCode	A logical value indicating whether to use the selected code in RStudio. Default is TRUE.
verbose	Logical flag to indicate whether to display the generated text. Default is TRUE.

Details

Add Roxygen Description to R Function

Value

A message indicating completion if 'SelectedCode' is TRUE, otherwise the Roxygen-annotated code is copied to the clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
addRoxygenDescription(Model = "gpt-4-0613", SelectedCode = FALSE)

## End(Not run)
```

autocreateFunction4R *autocreateFunction4R (development / experimental)*

Description

This function generates an R function based on a given description, proposes improvements, and then generates an improved version of the function. It is expected to use an AI model (possibly GPT-3 or similar) to perform these tasks. This is an experimental function.

Usage

```
autocreateFunction4R(
  Func_description,
  packages = "base",
  max_tokens = 250,
  View = TRUE,
  roxygen = TRUE,
  api_key = Sys.getenv("OPENAI_API_KEY"),
  verbose = TRUE
)
```

Arguments

Func_description	A character string that describes the function to be generated.
packages	A character string that specifies the packages to be used in the function. Default is "base".
max_tokens	An integer that specifies the maximum number of tokens to be returned by the AI model. Default is 250.
View	A logical that indicates whether to view the intermediate steps. Default is TRUE.
roxygen	A logical that indicates whether to include roxygen comments in the generated function. Default is TRUE.
api_key	A character string that represents the API key for the AI model being used. Default is the "OPENAI_API_KEY" environment variable.
verbose	A logical flag to print the message Default is TRUE.

Details

Generate and Improve R Functions

Value

The function returns a character string that represents the generated and improved R function.

Author(s)

Satoshi Kume

Examples

```
## Not run:
  Sys.setenv(OPENAI_API_KEY = "<APIKEY>")
  autocreateFunction4R(Func_description = "2*n+3 sequence")

## End(Not run)
```

---

chat4R *Chat4R: Interact with GPT-3.5 (default) using OpenAI API*

Description

This function uses the OpenAI API to interact with the GPT-3.5 model (default) and generates responses based on user input.

Usage

```
chat4R(
  content,
  Model = "gpt-3.5-turbo-16k",
  temperature = 1,
  simple = TRUE,
  fromJSON_parsed = FALSE,
  api_key = Sys.getenv("OPENAI_API_KEY")
)
```

Arguments

content	A string containing the user's input message.
Model	A string specifying the GPT model to use (default: "gpt-3.5-turbo-16k").
temperature	A numeric value controlling the randomness of the model's output (default: 1).
simple	Logical, if TRUE, only the content of the model's message will be returned.
fromJSON_parsed	Logical, if TRUE, content will be parsed from JSON.
api_key	A string containing the user's OpenAI API key. Defaults to the value of the environment variable "OPENAI_API_KEY".

Details

Chat4R Function

Value

A data frame containing the response from the GPT model.

Author(s)

Satoshi Kume

Examples

```
## Not run:
Sys.setenv(OPENAI_API_KEY = "Your API key")
response <- chat4R(content = "What is the capital of France?")
response

## End(Not run)
```

`chat4R_history` *chat4R_history: Use chat history for interacting with GPT.*

Description

This function use the chat history with the the specified GPT model, and chat the AI model.

Usage

```
chat4R_history(
  history,
  api_key = Sys.getenv("OPENAI_API_KEY"),
  Model = "gpt-3.5-turbo-16k",
  temperature = 1
)
```

Arguments

`history`	A list of message objects. Each object should have a 'role' that can be 'system', 'user', or 'assistant', and 'content' which is the content of the message from that role.
`api_key`	A string. Input your OpenAI API key. Defaults to the value of the environment variable "OPENAI_API_KEY".
`Model`	A string. The model to use for the chat completion. Default is "gpt-3.5-turbo-16k".
`temperature`	The temperature to use for the chat completion. Default is 1.

Details

Chat History for R

Value

A data frame containing the parsed response from the Web API server.

Author(s)

Satoshi Kume

Examples

```
## Not run:
Sys.setenv(OPENAI_API_KEY = "Your API key")

history <- list(list('role' = 'system', 'content' = 'You are a helpful assistant.'),
                list('role' = 'user', 'content' = 'Who won the world series in 2020?'))

chat4R_history(history)

## End(Not run)
```

`chatAI4pdf` *chatAI4pdf*

Description

Reads a PDF file and summarizes its content using a specified Large Language Model (LLM).

Usage

```
chatAI4pdf(pdf_file_path, nch = 2000, verbose = TRUE)
```

Arguments

`pdf_file_path`	The path to the PDF file to be summarized. Must be a string.
`nch`	The maximum number of characters for the summary. Default is 2000.
`verbose`	Logical flag to print the summary. Default is TRUE.

Details

Summarize PDF Text via LLM

This function reads a PDF file and summarizes its content using a specified Large Language Model (LLM).

Value

A string containing the summarized text.

Author(s)

Satoshi Kume

Examples

```
## Not run:

#Baktash et al: GPT-4: A REVIEW ON ADVANCEMENTS AND OPPORTUNITIES IN NATURAL LANGUAGE PROCESSING
pdf_file_path <- "https://arxiv.org/pdf/2305.03195.pdf"

#Execute
summary_text <- chatAI4pdf(pdf_file_path)

## End(Not run)
```

`checkErrorDet` *Check Error Details*

Description

A function to analyze and provide guidance on how to fix an error message copied from the R console.

Usage

```
checkErrorDet(
  Summary_nch = 100,
  Model = "gpt-4-0613",
  language = "English",
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

`Summary_nch`	An integer specifying the maximum number of characters for the summary.
`Model`	A string specifying the model to be used, default is "gpt-4-0314". Currently, "gpt-4", "gpt-4-0314" and "gpt-4-0613" can be selected as gpt-4 models. Execution with GPT-4 is recommended.
`language`	A string specifying the output language, default is "English".
`verbose`	A logical value to control the verbosity of the output, default is TRUE.
`SlowTone`	A logical value to control the printing speed of the output, default is FALSE.

Details

Check Error Details

This function provides a way to check error details in R. It takes an error message from the R console, executes the function, and shows how to fix the error in the specified language.

Value

The function prints the guidance on how to fix the error message.

Author(s)

Satoshi Kume

Examples

```
## Not run:
  # Copy the error message that you want to fix.
  checkErrorDet()
  checkErrorDet(language = "Japanese")

## End(Not run)
```

`checkErrorDet_JP` *Check Error Details in Japanese*

Description

A function to analyze and provide guidance on how to fix an error message copied from the R console.

Usage

```
checkErrorDet_JP(
  Summary_nch = 100,
  Model = "gpt-4-0613",
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

`Summary_nch`	An integer specifying the maximum number of characters for the summary.
`Model`	A string specifying the model to be used, default is "gpt-4-0314". Currently, "gpt-4", "gpt-4-0314" and "gpt-4-0613" can be selected as gpt-4 models. Execution with GPT-4 is recommended.
`verbose`	A logical value to control the verbosity of the output, default is TRUE.
`SlowTone`	A logical value to control the printing speed of the output, default is FALSE.

Details

Check Error Details in Japanese via RStudio API

This function provides a way to check error details in R. It reads the error message from the clipboard, executes the function, and shows how to fix the error in Japanese.

Value

The function prints the guidance on how to fix the error message in Japanese. If verbose is FALSE, it returns the guidance as a string.

Author(s)

Satoshi Kume

Examples

```
## Not run:
  # Analyzing error message from the clipboard
 checkErrorDet_JP(Summary_nch = 100, Model = "gpt-4-0613", verbose = TRUE, SlowTone = FALSE)

## End(Not run)
```

`completions4R` *completions4R: Generate text using OpenAI completions API (Legacy)*

Description

This function sends a request to the OpenAI completions API (Legacy) to generate text based on the provided prompt and parameters.

Usage

```
completions4R(
  prompt,
  api_key = Sys.getenv("OPENAI_API_KEY"),
  Model = "davinci-002",
  max_tokens = 50,
  temperature = 1,
  simple = TRUE
)
```

Arguments

`prompt`	A string. The initial text that the model responds to.
`api_key`	A string. The API key for OpenAI. Defaults to the value of the environment variable "OPENAI_API_KEY".
`Model`	A string. The model ID to use, such as "davinci-002".
`max_tokens`	Integer. The maximum number of tokens to generate.
`temperature`	A numeric value to control the randomness of the generated text. A value close to 0 produces more deterministic output, while a higher value (up to 2) produces more random output.
`simple`	If TRUE, returns the generated text without newline characters. If FALSE, returns the full response from the API.

Value

The generated text or the full response from the API, depending on the value of ‘simple‘.

Author(s)

Satoshi Kume

Examples

```
## Not run:
#This is a legacy function
Sys.setenv(OPENAI_API_KEY = "Your API key")

prompt <- "Translate the following English text to French: 'Hello, world!'"

completions4R(prompt)

## End(Not run)
```

`conversation4R` *Conversation Interface for R*

Description

Interface to communicate with OpenAI's models using R, maintaining a conversation history and allowing for initialization of a new conversation.

Usage

```
conversation4R(
  message,
  api_key = Sys.getenv("OPENAI_API_KEY"),
  template = "",
  ConversationBufferWindowMemory_k = 2,
  Model = "gpt-3.5-turbo-16k",
  language = "English",
  initialization = FALSE,
  verbose = TRUE
)
```

Arguments

`message`	A string containing the message to be sent to the model.
`api_key`	A string containing the OpenAI API key. Default is retrieved from the system environment variable "OPENAI_API_KEY".
`template`	A string containing the template for the conversation. Default is an empty string.

ConversationBufferWindowMemory_k	An integer representing the conversation buffer window memory. Default is 2.
Model	A string representing the model to be used. Default is "gpt-3.5-turbo-16k".
language	A string representing the language to be used in the conversation. Default is "English".
initialization	A logical flag to initialize a new conversation. Default is FALSE.
verbose	A logical flag to print the conversation. Default is TRUE.

Details

Conversation Interface for R with OpenAI

This function provides an interface to communicate with OpenAI's models using R. It maintains a conversation history and allows for initialization of a new conversation.

Value

Prints the conversation if verbose is TRUE. No return value.

Author(s)

Satoshi Kume

Examples

```
## Not run:
conversation4R(message = "Hello, OpenAI!",
               api_key = "your_api_key_here",
               language = "English",
               initialization = TRUE)

## End(Not run)
```

`convertBullet2Sentence` *convertBullet2Sentence*

Description

Convert bullet points to sentences using OpenAI GPT model.

Usage

```
convertBullet2Sentence(
  Model = "gpt-4-0613",
  temperature = 1,
  verbose = TRUE,
  SpeakJA = FALSE,
  SelectedCode = TRUE
)
```

Arguments

Model	The OpenAI GPT model to use for text generation. Default is "gpt-4-0613".
temperature	The temperature parameter for text generation. Default is 1.
verbose	Logical flag to indicate whether to display progress. Default is TRUE.
SpeakJA	Logical flag to indicate whether to use Japanese speech output. Default is FALSE.
SelectedCode	Logical flag to indicate whether to use selected text in RStudio. Default is TRUE.

Details

Convert Bullet Points to Sentences

This function takes bullet points as input and converts them into sentences. The function uses the OpenAI GPT model for text generation to assist in the conversion. The function can either take the selected text from the RStudio environment or from the clipboard.

Value

Inserts the converted sentences into the RStudio editor if 'SelectedCode' is TRUE, otherwise writes to clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
convertBullet2Sentence(Model = "gpt-4-0613", SelectedCode = FALSE)

## End(Not run)
```

---

convertRscript2Function

convertRscript2Function

Description

Convert selected R script to an R function using LLM.

Usage

```
convertRscript2Function(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE
)
```

Arguments

Model	The OpenAI GPT model to use for text generation. Default is "gpt-4-0613".
SelectedCode	Logical flag to indicate whether to use selected code in RStudio. Default is TRUE.
verbose	A logical value indicating whether to print the result to the console, default is TRUE.

Details

Convert Selected R Script to R Function

This function takes a selected portion of an R script and converts it into an R function. The function uses the OpenAI GPT model for text generation to assist in the conversion. The function can either take the selected code from the RStudio environment or from the clipboard.

Value

Inserts the converted function into the RStudio editor if ‘SelectedCode‘ is TRUE, otherwise writes to clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
convertRscript2Function(Model = "gpt-4-0613", SelectedCode = F)

## End(Not run)
```

convertScientificLiterature
convertScientificLiterature

Description

Convert input text into scientific literature.

Usage

```
convertScientificLiterature(Model = "gpt-4-0613", SelectedCode = TRUE)
```

Arguments

Model	The OpenAI GPT model to use for text generation. Default is "gpt-4-0613".
SelectedCode	Logical flag to indicate whether to read the input from RStudio's active document. Default is TRUE.

Details

Convert to Scientific Literature

This function assists in converting the input text into scientific literature. It uses the OpenAI GPT model for text generation to assist in the conversion process. The function reads the input either from the RStudio active document or the clipboard.

Value

Inserts the converted text into the RStudio active document if SelectedCode is TRUE, otherwise writes to the clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
convertScientificLiterature(SelectedCode = FALSE)

## End(Not run)
```

createEBAYdes *Create eBay Product Description*

Description

This function generates a product description for eBay listings in English. It uses the GPT-4 model for text generation and can take input from either RStudio or the clipboard.

Usage

```
createEBAYdes(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Model	The GPT-4 model to use for text generation. Default is "gpt-4-0613".
SelectedCode	Whether to get the input from the selected code in RStudio. Default is TRUE.
verbose	Whether to display progress information. Default is TRUE.
SlowTone	Whether to print the output slowly. Default is FALSE.

Details

Create eBay Product Description

Value

The generated eBay product description.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
createEBAYdes(Model = "gpt-4-0613", SelectedCode = FALSE, verbose = TRUE)

## End(Not run)
```

createImagePrompt_v1 *Create Image Prompt version 1*

Description

This function creates a prompt for generating an image from text using an AI model. This is an experimental function.

Usage

```
createImagePrompt_v1(content, Model = "gpt-3.5-turbo-16k", len = 200)
```

Arguments

content	A character string describing the image to be generated. If not provided, the function will throw a warning and stop.
Model	A character string specifying the AI model to be used for text generation.
len	Integer specifying the maximum length of the text input.

Details

Create Image Prompt version 1

Value

A character string that serves as the prompt for generating an image.

Author(s)

Satoshi Kume

Examples

```
## Not run:
createImagePrompt_v1(content = "A Japanese girl animation with blonde hair.")

## End(Not run)
```

createImagePrompt_v2 *Generate Image Prompts version 2*

Description

Generates optimal prompts for creating images using the OpenAI API. Given a base prompt, image attributes, and model parameters, it generates the optimal prompts for image creation. This is an experimental function.

Usage

```
createImagePrompt_v2(
  Base_prompt = "",
  removed_from_image = "",
  stable_diffusion = "N/A",
  Model = "gpt-3.5-turbo-16k",
  len = 1000
)
```

Arguments

Base_prompt	A string. This is the base prompt that forms the basis for the generated prompts.
removed_from_image	A string. This is an attribute that should be removed from the image.
stable_diffusion	A string. This parameter is used to control the stability of the diffusion process.
Model	A string. This is the model used for generating the prompts. Default is "gpt-3.5-turbo-16k".
len	An integer. This is the maximum length of the generated prompts. Must be between 1 and 1000. Default is 1000.

Details

Generate Image Prompts version 2

Value

A vector of strings. Each string in the vector is a generated prompt.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Define the base prompt and image attributes
base_prompt = "A sunset over a serene lake"
removed_from_image = "The sun"
stable_diffusion = "Moderate"

# Generate image prompts
res <- createImagePrompt_v2(Base_prompt = base_prompt, removed_from_image = removed_from_image,
stable_diffusion = stable_diffusion, len = 100)

# Print the generated prompts
print(res)

## End(Not run)
```

---

createRcode *Create R Code from Clipboard Content and Output into the R Console*

Description

Reads text from the clipboard and generates R code based on the given input, printing the result to the R console.

Usage

```
createRcode(
  Summary_nch = 100,
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Summary_nch	The maximum number of characters for the summary.
Model	The model to be used for code generation, default is "gpt-4-0613".
SelectedCode	A logical flag to indicate whether to read from RStudio's selected text. Default is TRUE.
verbose	A logical value indicating whether to print the result to the console, default is TRUE.
SlowTone	A logical value indicating whether to print the result slowly, default is FALSE.

Details

Create R Code from Selected Text or Clipboard Content

This function reads text from your selected text or clipboard, interprets it as a prompt, and generates R code based on the given input. The generated R code is then printed to the R console with optional slow printing. This function can be executed from RStudio's Addins menu.

Value

Prints the generated R code to the R console.

RStudio Addins

This function can be added to RStudio's Addins menu for easy access.

Author(s)

Satoshi Kume

Examples

```
## Not run:

# Copy the origin of R code to your clipboard then execute from RStudio's Addins.


## End(Not run)
```

---

createRfunction *Create R Function from Selected Text or Clipboard Content and Output into the R Console*

Description

This function reads text either from your selected text in RStudio or from the clipboard, interprets it as a prompt, and generates an R function based on the given input. The generated R code is then printed into the source file or the R console with optional slow printing.

Usage

```
createRfunction(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Model	A character string representing the model to be used. Default is "gpt-4-0613".
SelectedCode	A logical value indicating if the selected text should be used as input. Default is TRUE.
verbose	A logical value indicating if progress should be printed. Default is TRUE.
SlowTone	A logical value indicating if slow printing should be used. Default is FALSE.

Details

Create R Function from Selected Text or Clipboard Content

Value

This function returns the generated R code as a clipboard content if SelectedCode is FALSE.

Author(s)

Satoshi Kume

Examples

```
## Not run:

#Copy the idea text of the R function to your clipboard and run this function.
createRfunction(SelectedCode = FALSE)

## End(Not run)
```

---

createSpecifications4R

Create Specifications for R Function

Description

This function generates specifications for an R function from your selected text or clipboard. It takes in a text input, model name, verbosity, and tone speed to generate the specifications.

Usage

```
createSpecifications4R(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Model	A character string specifying the GPT model to be used. Default is "gpt-4-0613".
SelectedCode	A logical flag to indicate whether to read from RStudio's selected text. Default is TRUE.
verbose	A logical value indicating whether to print the output. Default is TRUE.
SlowTone	A logical value indicating whether to print the output slowly. Default is FALSE.

Details

Create Specifications for R Function

Value

The function prints the generated specifications to the console.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
createSpecifications4R(input = "Your R function specification")

## End(Not run)
```

---

designPackage *designPackage*

Description

Assist in proposing the overall design and architecture of an R package.

Usage

```
designPackage(Model = "gpt-4-0613", verbose = TRUE, SlowTone = FALSE)
```

Arguments

Model	The OpenAI GPT model to use for text generation. Default is "gpt-4-0613".
verbose	Logical flag to indicate whether to display the generated text. Default is TRUE.
SlowTone	Logical flag to indicate whether to print the text slowly. Default is FALSE.

Details

Design Package for R

This function assists in proposing the overall design and architecture of an R package. It uses the OpenAI GPT model for text generation to assist in the design process. The function reads the input from the clipboard.

Value

Prints the proposed design and architecture based on the verbosity and tone speed settings.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Copy the text into your clipboard then execute
designPackage(Model = "gpt-4-0613", verbose = TRUE, SlowTone = FALSE)

## End(Not run)
```

---

discussion_flow_v1 *discussion_flow_v1: Interactions and Flow Control Between LLM-based Bots (LLBs)*

Description

Simulates interactions and flow control between three different roles of LLM-based bots (LLBs).

Usage

```
discussion_flow_v1(
  issue,
  Domain = "bioinformatics",
  Model = "gpt-4-0613",
  api_key = Sys.getenv("OPENAI_API_KEY"),
  language = "English",
  Summary_nch = 50,
  verbose = TRUE,
  Nonfuture = TRUE,
  sayENorJA = TRUE
)
```

Arguments

issue	The issue to be discussed. Example: "I want to solve linear programming and create a timetable."
Domain	The domain of the discussion, default is "bioinformatics".
Model	The model to be used, default is "gpt-4-0613".
api_key	The API key for OpenAI, default is retrieved from the system environment variable "OPENAI_API_KEY".
language	The language for the discussion, default is "English".
Summary_nch	The number of characters for the summary, default is 50.
verbose	Logical, whether to print verbose output, default is TRUE.
Nonfuture	Logical, whether to use an asynchronous processing or not, default is not to use (TRUE).
sayENorJA	Logical, whether to say in English or Japanese, default is TRUE. This feature is available on macOS system only.

Details

Interactions and Flow Control Between LLM-based Bots (LLBs)

This function is described to simulate the interactions and flow control between three different roles of LLM-based bots, abbreviated as LLBs, and to reproduce more realistic dialogues and discussions. Here is a brief description of the roles: A (Beginner): This bot generates questions and summaries based on the content of the discussion provided by the user. B (Expert): This bot provides professional answers to questions posed by LLB A. C (Peer Reviewer): This bot reviews the dialog between LLB A and LLB B and suggests improvements or refinements. The three parties independently call the OpenAI API according to their roles. In addition, it keeps track of the conversation history between the bots and performs processes such as questioning, answering, and peer review. The function is designed to work in a "domain," which is essentially a specific area or topic around which conversations revolve. It is recommended to use GPT-4 or a model with higher accuracy than GPT-4. English is recommended as the input language, but the review will also be conducted in Japanese, the native language of the author.

Value

A summary of the conversation between the bots.

Author(s)

Satoshi Kume

Examples

```
## Not run:
issue <-  "I want to solve linear programming and create a timetable."

#Run Discussion with the domain of bioinformatics
discussion_flow_v1(issue)

## End(Not run)
```

discussion_flow_v2 *discussion_flow_v2: Interactions and Flow Control Between LLM-based Bots (LLBs)*

Description

Simulates interactions and flow control between three different roles of LLM-based bots (LLBs).

Usage

```
discussion_flow_v2(
  issue,
  Domain = "bioinformatics",
  Model = "gpt-4-0613",
  api_key = Sys.getenv("OPENAI_API_KEY"),
  language = "English",
  Summary_nch = 50,
  Sentence_difficulty = 2,
  R_expert_setting = TRUE,
  verbose = TRUE,
  sayENorJA = TRUE,
  rep_x = 3
)
```

Arguments

issue	The issue to be discussed. Example: "I want to solve linear programming and create a timetable."
Domain	The domain of the discussion, default is "bioinformatics".
Model	The LLM model to be used, default is "gpt-4-0613".
api_key	The API key for OpenAI, default is retrieved from the system environment variable "OPENAI_API_KEY".
language	The language for the discussion, default is "English".
Summary_nch	The number of characters for the summary, default is 50.
Sentence_difficulty	Numeric, the complexity level for sentence construction, default is 2.
R_expert_setting	Logical, whether R expert settings are enabled, default is TRUE.
verbose	Logical, whether to print verbose output, default is TRUE.
sayENorJA	Logical, whether to speak in English or Japanese, default is TRUE. This feature is available on macOS systems only.
rep_x	Numeric, a number of repeats for the conversations, default is 3.

Details

Interactions and Flow Control Between LLM-based Bots (LLBs)

In the v2 model, we added a regulation of the difficulty of the sentence, the human intervention in their conversation between LLM bots, and number of repetitions of conversation. This function is described to simulate the interactions and flow control between three different roles of LLM-based bots, abbreviated as LLBs, namely A (Beginner), B (Expert), and C (Peer Reviewer). These roles have distinct functions and work together to facilitate more complex and meaningful discussions. Here is a brief description of the roles: A (Beginner): This bot generates questions and summaries based on the content of the discussion provided by the user. B (Expert): This bot provides professional answers to questions posed by LLB A. C (Peer Reviewer): This bot reviews the dialog between LLB A and LLB B and suggests improvements or refinements. The three parties independently call the OpenAI API according to their roles. In addition, it keeps track of the conversation history between the bots and performs processes such as questioning, answering, and peer review. The function is designed to work in a "domain," which is essentially a specific area or topic around which conversations revolve. It is recommended to use GPT-4 or a model with higher accuracy than GPT-4. English is recommended as the input language, but the review will also be conducted in Japanese, the native language of the author.

Value

A summary of the conversation between the bots.

Author(s)

Satoshi Kume

Examples

```
## Not run:
issue <-  "I want to solve linear programming and create a timetable."

#Run Discussion with the domain of bioinformatics
discussion_flow_v2(issue)

## End(Not run)
```

`enrichTextContent` *Enrich Text Content v2*

Description

This function doubles the amount of text without changing its meaning. The GPT-4 model is currently recommended for text generation. It can either read from the RStudio selection or the clipboard.

Usage

```
enrichTextContent(Model = "gpt-4", SelectedCode = TRUE, verbose = TRUE)
```

Arguments

Model	A character string specifying the AI model to be used for text enrichment. Default is "gpt-4".
SelectedCode	A logical flag to indicate whether to read from RStudio's selected text. Default is TRUE.
verbose	Logical flag to indicate whether to display the generated text. Default is TRUE.

Details

Enrich Text Content

Value

If SelectedCode is TRUE, the enriched text is inserted into the RStudio editor and a message "Finished!!" is returned. Otherwise, the enriched text is placed into the clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
enrichTextContent(Model = "gpt-4", SelectedCode = TRUE)

## End(Not run)
```

---

extractKeywords *extractKeywords*

Description

Extract keywords from input text and output them in a comma-separated format.

Usage

```
extractKeywords(Model = "gpt-4-0613", verbose = TRUE, SlowTone = FALSE)
```

Arguments

Model	The OpenAI GPT model to use for text generation. Default is "gpt-4-0613".
verbose	Logical flag to indicate whether to print the result. Default is TRUE.
SlowTone	Logical flag to indicate the speed of the output. Default is FALSE.

Details

Extract Keywords from Text

This function extracts keywords from the input text. It uses the OpenAI GPT model for text generation to assist in the extraction process. The function reads the input from the clipboard and outputs the extracted keywords in a comma-separated format.

Value

Prints the extracted keywords based on verbosity and tone speed settings.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
extractKeywords(Model = "gpt-4-0613", verbose = TRUE, SlowTone = FALSE)

## End(Not run)
```

---

ngsub *ngsub*

Description

Remove extra spaces and newline characters from text.

Usage

```
ngsub(text)
```

Arguments

`text`	The input text from which extra spaces and newline characters need to be removed.

Details

Remove Extra Spaces and Newline Characters

This function removes extra spaces and newline characters from the given text. It replaces sequences of multiple spaces with a single space and removes newline characters followed by a space.

Value

Returns the modified text as a character string.

Author(s)

Satoshi Kume

Examples

```
## Not run:
ngsub("This is  a text \n with  extra   spaces.")

## End(Not run)
```

`OptimizeRcode` *Optimize and Complete R Code*

Description

Optimizes and completes the R code from the selected code or clipboard.

Usage

```
OptimizeRcode(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  verbose_Reasons4change = FALSE,
  SlowTone = FALSE
)
```

Arguments

`Model`	A string specifying the machine learning model to use for code optimization. Default is "gpt-4-0613".
`SelectedCode`	A boolean indicating whether to get the code from RStudio or the clipboard. Default is TRUE.
`verbose`	A logical value indicating whether to print the result to the console, default is TRUE.
`verbose_Reasons4change`	A boolean indicating whether to provide detailed reasons for the changes made. Default is FALSE
`SlowTone`	A boolean indicating whether to print the output slowly. Default is FALSE.

Details

Optimize and Complete R Code

This function takes a snippet of R code and optimizes it for performance and readability. It uses a machine learning model to generate the optimized code.

Value

A message indicating the completion of the optimization process.

Author(s)

Satoshi Kume

Examples

```
## Not run:
#Copy your R code then run the following function.
OptimizeRcode(SelectedCode = FALSE)

## End(Not run)
```

---

proofreadEnglishText *Proofread English Text*

Description

A function to proofread English text or text in different languages during R package development. It translates the input into English if necessary and returns meticulously checked English text.

Usage

```
proofreadEnglishText(Model = "gpt-4", SelectedCode = TRUE, verbose = TRUE)
```

Arguments

Model	A string specifying the model to be used for proofreading, defaulting to "gpt-4-0314". Currently, "gpt-4", "gpt-4-0314" and "gpt-4-0613" can be selected as gpt-4 models. Execution with GPT-4 is recommended.
SelectedCode	A logical value indicating whether to read the selected text from the RStudio editor (TRUE) or from the clipboard (FALSE). Defaults to TRUE.
verbose	Logical flag to print the progress. Default is TRUE.

Details

Proofread English Text During R Package Development via RStudio API

This function provides a feature to proofread English text during the development of an R package. It can either take the selected text from the RStudio editor or read from the clipboard, executes the proofreading, and returns the result to the user's clipboard or replaces the selected text. The user can then paste and check the result if read from the clipboard. The function adheres to R package policies and carefully proofreads the English text. Execution with GPT-4 is recommended.

Value

The proofread text, which is also written to the clipboard if SelectedCode is FALSE, or replaces the selected text if SelectedCode is TRUE.

Author(s)

Satoshi Kume

Examples

```
## Not run:
  # Proofreading selected text in RStudio
  proofreadEnglishText(Model = "gpt-4-0613", SelectedCode = TRUE)
  # Proofreading text from the clipboard
  proofreadEnglishText(Model = "gpt-4-0613", SelectedCode = FALSE)

## End(Not run)
```

---

proofreadText *proofreadText*

Description

Proofreads text during the development of an R package.

Usage

```
proofreadText(Model = "gpt-4-0613", SelectedCode = TRUE, verbose = TRUE)
```

Arguments

Model	The Large Language Model to be used for proofreading. Default is "gpt-4-0613".
SelectedCode	Logical flag to indicate whether to use the selected text in RStudio editor. Default is TRUE.
verbose	Logical flag to print the progress. Default is TRUE.

Details

Proofread Text During R Package Development Through the RStudio API

This function offers a feature for proofreading text while developing an R package. It can either use the text selected in the RStudio editor or read from the clipboard, perform the proofreading, and then either replace the selected text or return the result to the user's clipboard. The language of the output will match the language of the input text. Using GPT-4 for execution is recommended.

Value

NULL if 'SelectedCode' is TRUE, otherwise returns the proofread text to the clipboard.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Proofread text from clipboard
proofreadText(SelectedCode = FALSE)

## End(Not run)
```

RcodeImprovements *Suggest Improvements to the R Code on Your Clipboard*

Description

This function uses LLM to analyze the R code from the clipboard and suggests improvements. The function can also control the verbosity and speed of the output.

Usage

```
RcodeImprovements(
  Summary_nch = 100,
  Model = "gpt-4-0613",
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Summary_nch	An integer specifying the maximum number of characters for the summary. Default is 100.
Model	A character string specifying the GPT model to be used. Default is "gpt-4-0613".
verbose	A logical value indicating whether to print the result. Default is TRUE.
SlowTone	A logical value indicating whether to print the result slowly. Default is FALSE.

Details

Suggest Improvements for R Code

Value

No return value; the function prints the suggestions for code improvement.

Author(s)

Satoshi Kume

Examples

```
## Not run:
#Copy your function to your clipboard
RcodeImprovements(Summary_nch = 100, Model = "gpt-4-0613")

## End(Not run)
```

---

removeQuotations *Remove All Types of Quotations from Text*

Description

This function takes a text string as input and removes all occurrences of single, double, and back quotations marks.

Usage

```
removeQuotations(text)
```

Arguments

text	A character string from which quotations will be removed.

Value

A character string with all types of quotations removed.

Author(s)

Satoshi Kume

Examples

```
## Not run:
removeQuotations("\"XXX'`\"YYY'`") # Returns "XXXYYY"

## End(Not run)
```

revisedText *Revised Scientific Text*

Description

This function prompts the user to input text, revision comments, and additional background information. It then revises the text according to the comments and outputs the revised text.

Usage

```
revisedText(verbose = TRUE)
```

Arguments

verbose	Logical, whether to output verbose messages, default is TRUE.

Details

Revise Scientific Text

Value

Revised text or relevant message.

Author(s)

Satoshi Kume

Examples

```
## Not run:
 revisedText()

## End(Not run)
```

searchFunction *Search R Functions based on Text*

Description

Searches for an R function related to the provided text either through the RStudio editor selection or clipboard.

Usage

```
searchFunction(
  Summary_nch = 100,
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Summary_nch	Numeric, number of characters to limit the function description (default = 100).
Model	String, the model used for the search, default is "gpt-4-0613".
SelectedCode	Logical, whether to get text from RStudio selection (default = TRUE).
verbose	Logical, whether to print the results verbosely (default = TRUE).
SlowTone	Logical, whether to slow down the print speed for readability (default = FALSE).

Details

Search the R function based on the provided text

This function searches for an R function that corresponds to the text provided either through the RStudio editor selection or the clipboard. It fetches the related R function and outputs its name, package, and a brief description. The function uses GPT-4 for its underlying search.

Value

Console output of the identified R function, its package, and a brief description.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# To search for an R function related to "linear regression"
searchFunction(Summary_nch = 50, SelectedCode = FALSE)

## End(Not run)
```

`slow_print_v2` *Slowly Print Text*

Description

Prints the characters of the input text string one by one, with a specified delay between each character. If the random parameter is set to TRUE, the delay will be a random value between 0.0001 and 0.3 seconds. Otherwise, the delay will be the value specified by the delay parameter.

Usage

```
slow_print_v2(text, random = FALSE, delay = 0.125)
```

Arguments

`text`	A string representing the text to be printed. Must be a non-NA string.
`random`	A logical value indicating whether the delay between characters should be random. Default is FALSE.
`delay`	A numeric value representing the fixed delay between characters in seconds. Default is 0.125. Must be a non-negative number.

Details

Slowly Print Text

This function prints the characters of a given text string one by one, with a specified delay between each character. The delay can be either fixed or random.

Value

Invisible NULL. The function prints the text to the console.

Author(s)

Satoshi Kume

Examples

```
## Not run:
slow_print_v2("Hello, World!")
slow_print_v2("Hello, World!", random = TRUE)
slow_print_v2("Hello, World!", delay = 0.1)

## End(Not run)
```

speakInJA *Speak Selected Text in Japanese*

Description

This function reads aloud the selected text in Japanese using the MacOS system's 'say' command. The function specifically uses the 'Kyoko' voice for the speech. It only works on MacOS systems.

Usage

```
speakInJA()
```

Details

Speak Selected Text in Japanese on MacOS System

Value

A message indicating the completion of the speech is returned.

Author(s)

Satoshi Kume

Examples

```
## Not run:
  # Select some text in RStudio and then run the rstudio addins

## End(Not run)
```

speakInJA_v2 *Speak Clipboard in Japanese*

Description

This function reads aloud the clipboard content in Japanese using the MacOS system's 'say' command. The function specifically uses the 'Kyoko' voice for the speech. It only works on MacOS systems.

Usage

```
speakInJA_v2()
```

Details

Speak Clipboard in Japanese on MacOS System

Value

A message indicating the completion of the speech is returned.

Author(s)

Satoshi Kume

Examples

```
## Not run:
  # Copy some text into your clipboard in RStudio and then run the function
  speakInJA_v2()

## End(Not run)
```

`summaryWebScrapingText` *Text Summary via Web Scraping*

Description

Scrapes Google Search results for the provided query and summarizes the content.

Usage

```
summaryWebScrapingText(
  query = "LLM",
  t = "w",
  gl = "us",
  hl = "en",
  URL_num = 5,
  verbose = TRUE,
  translateJA = FALSE
)
```

Arguments

`query`	The search query. Default is "LLM".
`t`	Time period for search. 'w' for last week, 'm' for last month, 'y' for last year. Default is 'w'.
`gl`	Geographical location based on ISO 3166-1 alpha-2 country code. Default is 'us'.
`hl`	Language for search results based on ISO 639-1 language code. Default is 'en'.
`URL_num`	Number of URLs to scrape. Default is 5.
`verbose`	A boolean value indicating if details should be printed. Default is TRUE.
`translateJA`	A boolean value indicating if results should be translated to Japanese. Default is FALSE.

Details

Summarize Text via Web Scraping of Google Search

Scrape text information from Google Search and summarize it using LLM. Uses OpenAI API key for execution. Translation to Japanese requires a Deepl API key.

Value

Returns a list of summaries.

Author(s)

Satoshi Kume

Examples

```
## Not run:
summaryWebScrapingText(query = "LLM", t = "w", gl = "us", hl = "en", URL_num = 5)

## End(Not run)
```

---

supportIdeaGeneration *supportIdeaGeneration: Support Idea Generation from Selected Text or Clipboard.*

Description

Assist in generating ideas or concepts.

Usage

```
supportIdeaGeneration(
  Model = "gpt-4-0613",
  SelectedCode = TRUE,
  verbose = TRUE,
  SlowTone = FALSE
)
```

Arguments

Model	The OpenAI GPT model to use for text generation. Default is "gpt-4-0613".
SelectedCode	Logical flag to indicate whether to use the selected text in RStudio editor. Default is TRUE.
verbose	Logical flag to indicate whether to display the generated text. Default is TRUE.
SlowTone	Logical flag to indicate whether to print the text slowly. Default is FALSE.

Details

Support Idea Generation from Selected Text or Clipboard Input

This feature helps you generate ideas or concepts based on input from your selected text or clipboard. It uses the OpenAI GPT model for text generation to assist in the idea generation process. The function reads the input from the clipboard.

Value

Prints the generated ideas or concepts based on the verbosity and tone speed settings.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
supportIdeaGeneration()

## End(Not run)
```

---

textEmbedding *Text Embedding from OpenAI Embeddings API*

Description

This function calls the OpenAI Embeddings API to get the multidimensional vector via text embedding of the input text. This function uses the 'text-embedding-ada-002' model.

Usage

```
textEmbedding(text, api_key = Sys.getenv("OPENAI_API_KEY"))
```

Arguments

text	A string. The input text to get the embedding for. This should be a character string.
api_key	A string. The API key for the OpenAI API. Defaults to the value of the environment variable "OPENAI_API_KEY".

Value

A vector representing the text embeddings.

Author(s)

Satoshi Kume

Examples

```
## Not run:
Sys.setenv(OPENAI_API_KEY = "Your API key")
textEmbedding("Hello, world!")

## End(Not run)
```

TextSummary *Summarize Long Text*

Description

This function summarizes a long text using LLM. The development of this function started with the idea that it might be interesting to perform a copy-and-paste, sentence summarization and aims to be an evangelist for copy-and-paste LLM execution. It is recommended to run this function with GPT-4, but it is not cost effective and slow. This is still an experimental feature.

Usage

```
TextSummary(
  text = clipr::read_clip(),
  nch = 2000,
  verbose = TRUE,
  returnText = FALSE
)
```

Arguments

text	A character vector containing the text to be summarized. If not provided, the function will attempt to read from the clipboard.
nch	Integer specifying the number of characters at which to split the input text for processing.
verbose	A logical flag to print the message. Default is TRUE.
returnText	A logical flag to return summarized text results. Default is FALSE.

Details

Summarize Long Text

Value

The summarized text is placed into the clipboard and the function returns the result of `clipr::write_clip`.

Author(s)

Satoshi Kume

Examples

```
## Not run:
TextSummary(text = c("This is a long text to be summarized.",
                     "It spans multiple sentences and goes into much detail."),
            nch = 10)

## End(Not run)
```

TextSummaryAsBullet *Summarize Text into Bullet Points*

Description

This function takes a text input and summarizes it into a specified number of bullet points. It can either take the selected code from RStudio or read from the clipboard. The results are output to your clipboard.

Usage

```
TextSummaryAsBullet(
  Model = "gpt-4-0613",
  temperature = 1,
  verbose = TRUE,
  SelectedCode = TRUE
)
```

Arguments

Model	A string specifying the machine learning model to use for text summarization. Default is "gpt-4-0613".
temperature	A numeric value between 0 and 1 indicating the randomness of the text generation. Default is 1.
verbose	A logical value indicating whether to print the summary. Default is FALSE.
SelectedCode	A logical value indicating whether to use the selected code in RStudio. Default is TRUE.

Details

Summarize Selected Text into Bullet Points

Value

The summarized text in bullet points is returned.

Author(s)

Satoshi Kume

Examples

```
## Not run:
# Option 1
# Select some text in RStudio and then run the rstudio addins
# Option 2
# Copy the text into your clipboard then execute
TextSummaryAsBullet()

## End(Not run)
```

Index