

# Package ‘asa’

December 28, 2025

**Title** AI Search Agent for Large-Scale Research Automation

**Version** 0.1.0

**Description** Provides an LLM-powered research agent for performing AI search tasks at large scales. Uses a ReAct (Reasoning + Acting) agent pattern with web search capabilities via DuckDuckGo and Wikipedia. Implements DeepAgent-style memory folding for context management. The agent is built on 'LangGraph' and supports multiple LLM backends including 'OpenAI', 'Groq', and 'xAI'.

**URL** <https://github.com/cjerzak/asa-software>

**BugReports** <https://github.com/cjerzak/asa-software/issues>

**Depends** R (>= 4.0.0)

**License** GPL-3

**Encoding** UTF-8

**Imports** reticulate (>= 1.28), jsonlite, rlang, digest, processx

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, future, future.apply

**VignetteBuilder** knitr

**RoxygenNote** 7.3.3

**Config/testthat.edition** 3

**SystemRequirements** Python (>= 3.11), Conda, Tor (optional, for anonymous searching)

**NeedsCompilation** no

**Author** Connor Jerzak [aut, cre] (ORCID: <<https://orcid.org/0000-0003-1914-8905>>)

**Maintainer** Connor Jerzak <connor.jerzak@gmail.com>

## Contents

asa-package . . . . .	5
.asa_option . . . . .	5
.augment_prompt_temporal . . . . .	6
.build_trace . . . . .	6
.close_http_clients . . . . .	7
.create_agent . . . . .	7
.create_http_clients . . . . .	8
.create_llm . . . . .	8

.create_research_config . . . . .	9
.create_research_graph . . . . .	9
.create_tools . . . . .	9
.extract_fields . . . . .	10
.extract_json_from_trace . . . . .	10
.extract_json_object . . . . .	10
.extract_response_text . . . . .	11
.get_default_backend . . . . .	11
.get_default_conda_env . . . . .	11
.get_default_model . . . . .	11
.get_default_workers . . . . .	12
.get_extdata_path . . . . .	12
.get_local_ip . . . . .	12
.get_python_path . . . . .	13
.handle_response_issues . . . . .	13
.import_python_module . . . . .	13
.import_python_packages . . . . .	14
.import_research_modules . . . . .	14
.invoke_memory_folding_agent . . . . .	14
.invoke_standard_agent . . . . .	14
.is_initialized . . . . .	15
.normalize_schema . . . . .	15
.parse_json_response . . . . .	15
.process_research_results . . . . .	16
.resume_research . . . . .	16
.run_agent . . . . .	16
.run_research . . . . .	17
.run_research_with_progress . . . . .	17
.save_checkpoint . . . . .	17
.stop_validation . . . . .	18
.validate_api_key . . . . .	18
.validate_asa_agent . . . . .	18
.validate_asa_response . . . . .	19
.validate_asa_result . . . . .	19
.validate_build_backend . . . . .	19
.validate_build_prompt . . . . .	20
.validate_choice . . . . .	20
.validate_conda_env . . . . .	20
.validate_configure_search . . . . .	21
.validate_consistency . . . . .	21
.validate_dataframe . . . . .	22
.validate_initialize_agent . . . . .	22
.validate_logical . . . . .	23
.validate_positive . . . . .	23
.validate_process_outputs . . . . .	23
.validate_proxy_url . . . . .	24
.validate_range . . . . .	24
.validate_required . . . . .	24
.validate_research_inputs . . . . .	25
.validate_run_agent . . . . .	25
.validate_run_task . . . . .	25
.validate_run_task_batch . . . . .	26

.validate_s3_class . . . . .	26
.validate_string . . . . .	27
.validate_string_vector . . . . .	27
.validate_temporal . . . . .	28
.with_temporal . . . . .	28
as.data.frame.asa_audit_result . . . . .	29
as.data.frame.asa_enumerate_result . . . . .	29
as.data.frame.asa_result . . . . .	30
asa_agent . . . . .	30
ASA_API_ENDPOINTS . . . . .	31
ASA_API_KEY_ENV_VARS . . . . .	31
asa_audit . . . . .	31
asa_audit_result . . . . .	33
asa_config . . . . .	34
ASA_DEFAULT_BACKEND . . . . .	36
ASA_DEFAULT_BUDGET_QUERIES . . . . .	36
ASA_DEFAULT_BUDGET_TIME . . . . .	36
ASA_DEFAULT_BUDGET_TOKENS . . . . .	37
ASA_DEFAULT_CONDA_ENV . . . . .	37
ASA_DEFAULT_INTER_SEARCH_DELAY . . . . .	37
ASA_DEFAULT_MAX_RESULTS . . . . .	38
ASA_DEFAULT_MAX_RETRIES . . . . .	38
ASA_DEFAULT_MAX_ROUNDS . . . . .	38
ASA_DEFAULT_MEMORY_FOLDING . . . . .	39
ASA_DEFAULT_MEMORY_KEEP_RECENT . . . . .	39
ASA_DEFAULT_MEMORY_THRESHOLD . . . . .	39
ASA_DEFAULT_MODEL . . . . .	40
ASA_DEFAULT_NOVELTY_MIN . . . . .	40
ASA_DEFAULT_NOVELTY_WINDOW . . . . .	40
ASA_DEFAULT_PAGE_LOAD_WAIT . . . . .	41
ASA_DEFAULT_PLATEAU_ROUNDS . . . . .	41
ASA_DEFAULT_PROXY . . . . .	41
ASA_DEFAULT_RATE_LIMIT . . . . .	42
ASA_DEFAULT_TEMPERATURES . . . . .	42
ASA_DEFAULT_TIMEOUT . . . . .	42
ASA_DEFAULT_WIKI_CHARS . . . . .	43
ASA_DEFAULT_WIKI_TOP_K . . . . .	43
ASA_DEFAULT_WORKERS . . . . .	43
asa_enumerate . . . . .	44
asa_enumerate_result . . . . .	47
ASA_OUTPUT_FORMATS . . . . .	48
ASA_PRINT_WIDTH . . . . .	48
ASA_RATE_LIMIT_WAIT . . . . .	49
ASA_RECURRENCE_LIMIT_FOLDING . . . . .	49
ASA_RECURRENCE_LIMIT_STANDARD . . . . .	49
asa_response . . . . .	50
asa_result . . . . .	50
ASA_STATUS_ERROR . . . . .	51
ASA_STATUS_SUCCESS . . . . .	51
ASA_SUPPORTED_BACKENDS . . . . .	52
ASA_TEMPORAL_STRICTNESS . . . . .	52
ASA_TIME_FILTERS . . . . .	52

ASA_TRUNCATE_LENGTH . . . . .	53
build_backend . . . . .	53
build_prompt . . . . .	54
check_backend . . . . .	55
clean_whitespace . . . . .	55
configure_search . . . . .	56
configure_search_logging . . . . .	57
configure_temporal . . . . .	58
decode_html . . . . .	59
extract_agent_results . . . . .	59
extract_search_snippets . . . . .	60
extract_search_tiers . . . . .	61
extract_urls . . . . .	61
extract_wikipedia_content . . . . .	62
format_duration . . . . .	62
get_agent . . . . .	63
get_tor_ip . . . . .	63
initialize_agent . . . . .	64
is_tor_running . . . . .	66
json_escape . . . . .	66
print.asa_agent . . . . .	67
print.asa_audit_result . . . . .	67
print.asa_config . . . . .	68
print.asa_enumerate_result . . . . .	68
print.asa_response . . . . .	69
print.asa_result . . . . .	69
print.asa_search . . . . .	70
print.asa_temporal . . . . .	70
print2 . . . . .	71
process_outputs . . . . .	71
reset_agent . . . . .	72
rotate_tor_circuit . . . . .	72
run_task . . . . .	73
run_task_batch . . . . .	75
safe_json_parse . . . . .	76
search_options . . . . .	77
summary.asa_agent . . . . .	78
summary.asa_audit_result . . . . .	79
summary.asa_enumerate_result . . . . .	79
summary.asa_response . . . . .	80
summary.asa_result . . . . .	80
temporal_options . . . . .	81
truncate_string . . . . .	82
write_csv.asa_enumerate_result . . . . .	83

---

asa-package

*asa: AI Search Agent for Large-Scale Research Automation*

---

## Description

The `asa` package provides an LLM-powered research agent for performing AI search tasks at large scales using web search capabilities.

The agent uses a ReAct (Reasoning + Acting) pattern implemented via LangGraph, with tools for searching DuckDuckGo and Wikipedia. It supports multiple LLM backends (OpenAI, Groq, xAI) and implements DeepAgent-style memory folding for managing long conversations.

## Main Functions

- `build_backend`: Set up the Python conda environment
- `initialize_agent`: Initialize the search agent
- `run_task`: Run a structured task with the agent
- `run_task_batch`: Run multiple tasks in batch

## Configuration

The package requires a Python environment with LangChain and related packages. Use `build_backend` to create this environment automatically.

For anonymous searching, the package can use Tor as a SOCKS5 proxy. Install Tor via `brew install tor` (macOS) and start it with `brew services start tor`.

## Author(s)

**Maintainer:** Connor Jerzak <connor.jerzak@gmail.com> ([ORCID](#))

## See Also

Useful links:

- <https://github.com/cjerzak/asa-software>
- Report bugs at <https://github.com/cjerzak/asa-software/issues>

---

.asa\_option

*Get Package Option or Default*

---

## Description

Returns the value of an `asa` package option, or the default if not set. Options can be set via `options(asa.option_name = value)`.

## Usage

`.asa_option(name, default)`

**Arguments**

<code>name</code>	Option name (without "asa." prefix)
<code>default</code>	Default value if option not set

**Value**

Option value or default

---

**.augment\_prompt\_temporal**

*Augment Prompt with Temporal Context*

---

**Description**

Adds temporal date hints to the prompt when after/before dates are specified. This helps guide the agent to search for time-relevant information.

**Usage**

```
.augment_prompt_temporal(prompt, temporal)
```

**Arguments**

<code>prompt</code>	Original prompt
<code>temporal</code>	Temporal filtering list (may be NULL)

**Value**

Augmented prompt string

---

**.build\_trace**

*Build Trace from Raw Response*

---

**Description**

Build Trace from Raw Response

**Usage**

```
.build_trace(raw_response)
```

---

.close\_http\_clients     *Close HTTP Clients*

---

### Description

Safely closes the synchronous httpx client to prevent resource leaks. This is called automatically by reset\_agent() and when reinitializing.

### Usage

```
.close_http_clients()
```

### Details

Note: We no longer create or manage async clients from R (R-CRIT-001 fix). LangChain manages its own async client lifecycle internally.

### Value

Invisibly returns NULL

---

.create\_agent     *Create the LangGraph Agent*

---

### Description

Create the LangGraph Agent

### Usage

```
.create_agent(  
  llm,  
  tools,  
  use_memory_folding,  
  memory_threshold,  
  memory_keep_recent  
)
```

### Arguments

llm	LLM instance
tools	List of tools
use_memory_folding	Whether to use memory folding
memory_threshold	Messages before folding
memory_keep_recent	Messages to keep

`.create_http_clients`    *Create HTTP Client for API Calls*

## Description

Creates a synchronous httpx client for LLM API calls. Note: We intentionally do NOT create an async client. LangChain/OpenAI SDK creates its own async client internally when needed (for async operations). This avoids R-CRIT-001 where async client cleanup was unreliable from R since `aclose()` requires an async context.

## Usage

```
.create_http_clients(proxy, timeout)
```

## Arguments

<code>proxy</code>	Proxy URL or NULL
<code>timeout</code>	Timeout in seconds

## Value

A list with 'sync' client (async is NULL, letting LangChain manage it)

`.create_llm`    *Create LLM Instance*

## Description

Create LLM Instance

## Usage

```
.create_llm(backend, model, clients, rate_limit)
```

## Arguments

<code>backend</code>	Backend name
<code>model</code>	Model identifier
<code>clients</code>	HTTP clients (for OpenAI)
<code>rate_limit</code>	Requests per second

---

.create\_research\_config

*Create Research Configuration*

---

### Description

Create Research Configuration

### Usage

```
.create_research_config(  
    workers,  
    max_rounds,  
    budget,  
    stop_policy,  
    sources,  
    temporal = NULL  
)
```

---

.create\_research\_graph

*Create Research Graph*

---

### Description

Create Research Graph

### Usage

```
.create_research_graph(agent, config_dict)
```

---

.create\_tools

*Create Search Tools*

---

### Description

Create Search Tools

### Usage

```
.create_tools(proxy)
```

### Arguments

proxy	Proxy URL or NULL
-------	-------------------

---

`.extract_fields`      *Extract Specific Fields from Response*

---

**Description**

Extract Specific Fields from Response

**Usage**

```
.extract_fields(text, fields)
```

**Arguments**

text	Response text
fields	Character vector of field names to extract

---

`.extract_json_from_trace`      *Extract JSON from Agent Traces*

---

**Description**

Internal function to extract JSON data from raw agent traces.

**Usage**

```
.extract_json_from_trace(text)
```

**Arguments**

text	Raw trace text
------	----------------

**Value**

Parsed JSON data as a list, or NULL if no JSON found

---

`.extract_json_object`      *Extract JSON Object from Text*

---

**Description**

Extract JSON Object from Text

**Usage**

```
.extract_json_object(text)
```

**Arguments**

text	Response text
------	---------------

---

.extract\_response\_text

*Extract Response Text from Raw Response*

---

### Description

Extract Response Text from Raw Response

### Usage

.extract\_response\_text(raw\_response, backend)

---

.get\_default\_backend

*Get Default Backend*

---

### Description

Get Default Backend

### Usage

.get\_default\_backend()

---

.get\_default\_conda\_env

*Get Default Conda Environment*

---

### Description

Get Default Conda Environment

### Usage

.get\_default\_conda\_env()

---

.get\_default\_model

*Get Default Model*

---

### Description

Get Default Model

### Usage

.get\_default\_model()

---

`.get_default_workers`    *Get Default Workers*

---

**Description**

Get Default Workers

**Usage**

```
.get_default_workers()
```

---

`.get_extdata_path`    *Get External Data Path*

---

**Description**

Returns the path to the package's external data directory.

**Usage**

```
.get_extdata_path(filename = NULL)
```

**Arguments**

`filename`              Optional filename within extdata directory

**Value**

Character string with the path

---

`.get_local_ip`              *Get Local IP Address (Cross-Platform)*

---

**Description**

Returns the local IP address for use with Exo backend. Works on Windows, macOS, and Linux.

**Usage**

```
.get_local_ip()
```

**Value**

Character string with the local IP address, or "127.0.0.1" on failure.

---

.get\_python\_path      *Get Package Python Module Path*

---

### Description

Returns the path to the Python modules shipped with the package.

### Usage

```
.get_python_path()
```

### Value

Character string with the path to inst/python

---

.handle\_response\_issues      *Handle Response Issues (Rate Limiting, Timeouts)*

---

### Description

Handle Response Issues (Rate Limiting, Timeouts)

### Usage

```
.handle_response_issues(trace, verbose)
```

---

.import\_python\_module      *Import Python Module into asa\_env*

---

### Description

Generic helper for importing Python modules from inst/python. Handles caching, path resolution, and error handling.

### Usage

```
.import_python_module(module_name, env_name = module_name, required = TRUE)
```

### Arguments

module_name	Name of the Python module (without .py)
env_name	Name in asa_env (defaults to module_name)
required	If TRUE, error on failure; if FALSE, return NULL

### Value

The imported Python module (invisibly), or NULL on failure if not required

---

```
.import_python_packages
```

*Import Required Python Packages*

---

**Description**

Import Required Python Packages

**Usage**

```
.import_python_packages()
```

---

```
.import_research_modules
```

*Import Research Python Modules*

---

**Description**

Import Research Python Modules

**Usage**

```
.import_research_modules()
```

---

```
.invoke_memory_folding_agent
```

*Invoke Memory Folding Agent*

---

**Description**

Invoke Memory Folding Agent

**Usage**

```
.invoke_memory_folding_agent(python_agent, prompt, recursion_limit)
```

---

```
.invoke_standard_agent
```

*Invoke Standard Agent*

---

**Description**

Invoke Standard Agent

**Usage**

```
.invoke_standard_agent(python_agent, prompt, recursion_limit)
```

---

.is\_initialized      *Check if ASA Agent is Initialized*

---

### Description

Check if ASA Agent is Initialized

### Usage

.is\_initialized()

### Value

Logical indicating if the agent has been initialized

---

.normalize\_schema      *Normalize Schema Input*

---

### Description

Normalize Schema Input

### Usage

.normalize\_schema(schema, query, verbose)

---

.parse\_json\_response      *Parse JSON Response*

---

### Description

Parse JSON Response

### Usage

.parse\_json\_response(response\_text)

### Arguments

response\_text    Response text from agent

---

`.process_research_results`  
*Process Research Results*

---

**Description**

Process Research Results

**Usage**

`.process_research_results(result, schema_dict, include_provenance)`

---

`.resume_research`      *Resume Research from Checkpoint*

---

**Description**

Resume Research from Checkpoint

**Usage**

`.resume_research(checkpoint_file, verbose)`

---

`.run_agent`      *Run the ASA Agent (Internal)*

---

**Description**

Internal function that invokes the search agent with a prompt. Users should use `run_task` instead.

**Usage**

```
.run_agent(
  prompt,
  agent = NULL,
  temporal = NULL,
  recursion_limit = NULL,
  verbose = FALSE
)
```

**Arguments**

<code>prompt</code>	The prompt to send to the agent
<code>agent</code>	An <code>asa_agent</code> object
<code>temporal</code>	Named list for temporal filtering
<code>recursion_limit</code>	Maximum number of agent steps
<code>verbose</code>	Print status messages

**Value**

An object of class `asa_response`

---

.run\_research

*Run Research (Non-Streaming)*

---

**Description**

Run Research (Non-Streaming)

**Usage**

```
.run_research(graph, query, schema_dict, config_dict)
```

---

.run\_research\_with\_progress

*Run Research with Progress Updates*

---

**Description**

Run Research with Progress Updates

**Usage**

```
.run_research_with_progress(  
    graph,  
    query,  
    schema_dict,  
    config_dict,  
    checkpoint_file,  
    verbose  
)
```

---

.save\_checkpoint

*Save Checkpoint*

---

**Description**

Save Checkpoint

**Usage**

```
.save_checkpoint(result, query, schema_dict, config_dict, checkpoint_file)
```

---

<code>.stop_validation</code>	<i>Stop with Formatted Validation Error</i>
-------------------------------	---

---

### Description

Creates a standardized error message with Got/Fix sections.

### Usage

```
.stop_validation(param_name, requirement, actual = NULL, fix = NULL)
```

### Arguments

<code>param_name</code>	Name of the parameter that failed validation
<code>requirement</code>	What the parameter should be
<code>actual</code>	What was actually received (optional, auto-formatted)
<code>fix</code>	Actionable fix suggestion

---

<code>.validate_api_key</code>	<i>Validate API Key for Backend</i>
--------------------------------	-------------------------------------

---

### Description

Checks that the required API key environment variable is set for the specified backend. Throws an informative error if missing.

### Usage

```
.validate_api_key(backend)
```

### Arguments

<code>backend</code>	LLM backend name
----------------------	------------------

### Value

Invisibly returns TRUE if valid

---

<code>.validate_asa_agent</code>	<i>Validate S3 Constructor: asa_agent</i>
----------------------------------	---

---

### Description

Validate S3 Constructor: asa\_agent

### Usage

```
.validate_asa_agent(python_agent, backend, model, config)
```

---

.validate\_asa\_response

*Validate S3 Constructor: asa\_response*

---

## Description

Validate S3 Constructor: asa\_response

## Usage

```
.validate_asa_response(  
    message,  
    status_code,  
    raw_response,  
    trace,  
    elapsed_time,  
    fold_count,  
    prompt  
)
```

---

.validate\_asa\_result

*Validate S3 Constructor: asa\_result*

---

## Description

Validate S3 Constructor: asa\_result

## Usage

```
.validate_asa_result(prompt, message, parsed, raw_output, elapsed_time, status)
```

---

.validate\_build\_backend

*Validate build\_backend() Parameters*

---

## Description

Validate build\_backend() Parameters

## Usage

```
.validate_build_backend(conda_env, conda, python_version)
```

---

`.validate_build_prompt`

*Validate build\_prompt() Parameters*

---

### Description

Validate build\_prompt() Parameters

### Usage

```
.validate_build_prompt(template)
```

---

`.validate_choice`

*Validate Choice from Set*

---

### Description

Validate Choice from Set

### Usage

```
.validate_choice(x, param_name, choices)
```

### Arguments

x	Value to check
param_name	Name for error message
choices	Valid choices

---

`.validate_conda_env`

*Validate Conda Environment Name*

---

### Description

Validate Conda Environment Name

### Usage

```
.validate_conda_env(x, param_name)
```

### Arguments

x	Value to check
param_name	Name for error message

---

`.validate_configure_search`

*Validate configure\_search() Parameters*

---

### Description

Validate configure\_search() Parameters

### Usage

```
.validate_configure_search(  
    max_results,  
    timeout,  
    max_retries,  
    retry_delay,  
    backoff_multiplier,  
    captcha_backoff_base,  
    page_load_wait,  
    inter_search_delay,  
    conda_env  
)
```

---

`.validate_consistency` *Validate Logical Consistency Between Parameters*

---

### Description

Validate Logical Consistency Between Parameters

### Usage

```
.validate_consistency(condition, message, fix)
```

### Arguments

condition	Condition that must be TRUE
message	Error message if condition is FALSE
fix	How to fix the issue

`.validate_dataframe`     *Validate Data Frame with Required Columns*

## Description

Validate Data Frame with Required Columns

## Usage

```
.validate_dataframe(x, param_name, required_cols = NULL)
```

## Arguments

<code>x</code>	Value to check
<code>param_name</code>	Name for error message
<code>required_cols</code>	Required column names (optional)

`.validate_initialize_agent`  
*Validate initialize\_agent() Parameters*

## Description

Validate initialize\_agent() Parameters

## Usage

```
.validate_initialize_agent(  

  backend,  

  model,  

  conda_env,  

  proxy,  

  use_memory_folding,  

  memory_threshold,  

  memory_keep_recent,  

  rate_limit,  

  timeout,  

  verbose  

)
```

---

.validate\_logical      *Validate Boolean*

---

### Description

Validate Boolean

### Usage

.validate\_logical(x, param\_name)

### Arguments

x	Value to check
param_name	Name for error message

---

.validate\_positive      *Validate Positive Number*

---

### Description

Validate Positive Number

### Usage

.validate\_positive(x, param\_name, allow\_zero = FALSE, integer\_only = FALSE)

### Arguments

x	Value to check
param_name	Name for error message
allow_zero	Allow zero values (default: FALSE)
integer_only	Require integer values (default: FALSE)

---

.validate\_process\_outputs  
    *Validate process\_outputs() Parameters*

---

### Description

Validate process\_outputs() Parameters

### Usage

.validate\_process\_outputs(df, parallel, workers)

`.validate_proxy_url`     *Validate URL Format (SOCKS5 Proxy)*

### Description

Validate URL Format (SOCKS5 Proxy)

### Usage

```
.validate_proxy_url(x, param_name)
```

### Arguments

<code>x</code>	Value to check (NULL is valid = no proxy)
<code>param_name</code>	Name for error message

`.validate_range`     *Validate Range*

### Description

Validate Range

### Usage

```
.validate_range(x, param_name, min = NULL, max = NULL)
```

### Arguments

<code>x</code>	Value to check (must already be validated as numeric)
<code>param_name</code>	Name for error message
<code>min</code>	Minimum allowed value (optional)
<code>max</code>	Maximum allowed value (optional)

`.validate_required`     *Validate Required Argument Presence*

### Description

Validate Required Argument Presence

### Usage

```
.validate_required(x, param_name)
```

### Arguments

<code>x</code>	Value to check
<code>param_name</code>	Name for error message

---

```
.validate_research_inputs
```

*Validate Research Inputs*

---

**Description**

Validate Research Inputs

**Usage**

```
.validate_research_inputs(  
    query,  
    schema,  
    output,  
    workers,  
    max_rounds,  
    budget,  
    stop_policy,  
    sources,  
    checkpoint_dir,  
    resume_from  
)
```

---

```
.validate_run_agent      Validate run_agent() Parameters
```

---

**Description**

Validate run\_agent() Parameters

**Usage**

```
.validate_run_agent(prompt, agent, recursion_limit, verbose)
```

---

```
.validate_run_task       Validate run_task() Parameters
```

---

**Description**

Validate run\_task() Parameters

**Usage**

```
.validate_run_task(prompt, output_format, agent, verbose)
```

---

```
.validate_run_task_batch  
    Validate run_task_batch() Parameters
```

---

### Description

Validate run\_task\_batch() Parameters

### Usage

```
.validate_run_task_batch(  
    prompts,  
    output_format,  
    agent,  
    parallel,  
    workers,  
    progress  
)
```

---

```
.validate_s3_class      Validate S3 Class
```

---

### Description

Validate S3 Class

### Usage

```
.validate_s3_class(x, param_name, expected_class)
```

### Arguments

x	Value to check
param_name	Name for error message
expected_class	Expected S3 class name

---

`.validate_string`      *Validate Non-Empty String*

---

### Description

Validate Non-Empty String

### Usage

```
.validate_string(x, param_name, allow_empty = FALSE, allow_na = FALSE)
```

### Arguments

<code>x</code>	Value to check
<code>param_name</code>	Name for error message
<code>allow_empty</code>	Allow empty strings (default: FALSE)
<code>allow_na</code>	Allow NA values (default: FALSE)

---

---

`.validate_string_vector`      *Validate Character Vector (Non-Empty)*

---

### Description

Validate Character Vector (Non-Empty)

### Usage

```
.validate_string_vector(x, param_name, min_length = 1L)
```

### Arguments

<code>x</code>	Value to check
<code>param_name</code>	Name for error message
<code>min_length</code>	Minimum required length (default: 1)

`.validate_temporal`      *Validate Temporal Filtering Parameters*

## Description

Validates and normalizes temporal filtering parameters used by `run_task()` and `asa_enumerate()`. Returns a normalized list or NULL if input is NULL.

## Usage

```
.validate_temporal(temporal, param_name = "temporal")
```

## Arguments

<code>temporal</code>	Named list with temporal filtering options, or NULL
<code>param_name</code>	Name for error messages (default: "temporal")

## Value

Normalized temporal list or NULL

`.with_temporal`      *Apply Temporal Filtering for a Single Operation*

## Description

Internal helper that applies temporal filtering, runs a function, and restores the original setting. Used by `run_task()` and `run_task_batch()`.

## Usage

```
.with_temporal(temporal, fn)
```

## Arguments

<code>temporal</code>	Named list with temporal options (time_filter, after, before)
<code>fn</code>	Function to run with temporal filtering applied

## Value

Result of `fn()`

---

```
as.data.frame.asa_audit_result  
Convert asa_audit_result to Data Frame
```

---

**Description**

Convert asa\_audit\_result to Data Frame

**Usage**

```
## S3 method for class 'asa_audit_result'  
as.data.frame(x, ...)
```

**Arguments**

x	An asa_audit_result object
...	Additional arguments (ignored)

**Value**

The audited data.frame with audit columns

---

```
as.data.frame.asa_enumerate_result  
Convert asa_enumerate_result to Data Frame
```

---

**Description**

Convert asa\_enumerate\_result to Data Frame

**Usage**

```
## S3 method for class 'asa_enumerate_result'  
as.data.frame(x, ...)
```

**Arguments**

x	An asa_enumerate_result object
...	Additional arguments (ignored)

**Value**

The data data.frame from the result

---

`as.data.frame.asa_result`  
*Convert asa\_result to Data Frame*

---

**Description**

Convert asa\_result to Data Frame

**Usage**

```
## S3 method for class 'asa_result'
as.data.frame(x, ...)
```

**Arguments**

<code>x</code>	An asa_result object
<code>...</code>	Additional arguments (ignored)

**Value**

A single-row data frame

---

`asa_agent`                    *Constructor for asa\_agent Objects*

---

**Description**

Creates an S3 object representing an initialized ASA search agent.

**Usage**

```
asa_agent(python_agent, backend, model, config)
```

**Arguments**

<code>python_agent</code>	The underlying Python agent object
<code>backend</code>	LLM backend name (e.g., "openai", "groq")
<code>model</code>	Model identifier
<code>config</code>	Agent configuration list

**Value**

An object of class `asa_agent`

---

ASA_API_ENDPOINTS	<i>Backend API Endpoints</i>
-------------------	------------------------------

---

**Description**

Backend API Endpoints

**Usage**

ASA\_API\_ENDPOINTS

**Format**

An object of class `list` of length 3.

---

ASA_API_KEY_ENV_VARS	<i>Environment Variables for API Keys</i>
----------------------	---

---

**Description**

Environment Variables for API Keys

**Usage**

ASA\_API\_KEY\_ENV\_VARS

**Format**

An object of class `list` of length 5.

---

asa_audit	<i>Audit Enumeration Results for Completeness and Quality</i>
-----------	---

---

**Description**

Validates enumeration results for completeness, consistency, and data quality using either Claude Code (CLI) or a LangGraph-based audit pipeline.

## Usage

```
asa_audit(
  result,
  query = NULL,
  known_universe = NULL,
  checks = c("completeness", "consistency", "gaps", "anomalies"),
  backend = c("claude_code", "langgraph"),
  claude_model = "claude-sonnet-4-20250514",
  llm_model = "gpt-4.1-mini",
  interactive = FALSE,
  confidence_threshold = 0.8,
  timeout = 120,
  verbose = TRUE,
  agent = NULL
)
```

## Arguments

<code>result</code>	An <code>asa_enumerate_result</code> object or a <code>data.frame</code> to audit
<code>query</code>	The original enumeration query (inferred from <code>result</code> if <code>NULL</code> )
<code>known_universe</code>	Optional vector of expected items for completeness check
<code>checks</code>	Character vector of checks to perform. Options: "completeness", "consistency", "gaps", "anomalies". Default runs all checks.
<code>backend</code>	Backend to use for auditing: "claude_code" (CLI) or "langgraph"
<code>claude_model</code>	Model to use with Claude Code backend
<code>llm_model</code>	Model to use with LangGraph backend
<code>interactive</code>	If <code>TRUE</code> and using <code>claude_code</code> backend, spawn an interactive Claude Code session instead of programmatic invocation
<code>confidence_threshold</code>	Flag items with confidence below this threshold
<code>timeout</code>	Timeout in seconds for the audit operation
<code>verbose</code>	Print progress messages
<code>agent</code>	Existing <code>asa_agent</code> for LangGraph backend (optional)

## Details

The audit function adds three columns to the data:

- `_audit_flag`: "ok", "warning", or "suspect"
- `_audit_notes`: Explanation of any issues
- `_confidence_adjusted`: Revised confidence after audit

`## Audit Checks`

**completeness**: Checks for missing items by comparing against `known_universe` (if provided) or using domain knowledge.

**consistency**: Validates data types, patterns, and value ranges.

**gaps**: Identifies systematic patterns of missing data (geographic, temporal, categorical gaps).

**anomalies**: Detects duplicates, outliers, and suspicious patterns.

**Value**

An `asa_audit_result` object containing:

<code>data</code>	Original data with audit columns added ( <code>_audit_flag</code> , <code>_audit_notes</code> )
<code>audit_summary</code>	High-level summary of findings
<code>issues</code>	List of identified issues with severity and descriptions
<code>recommendations</code>	Suggested remediation queries
<code>completeness_score</code>	0-1 score for data completeness
<code>consistency_score</code>	0-1 score for data consistency

**Examples**

```
## Not run:
# Audit enumeration results with Claude Code
senators <- asa_enumerate(
  query = "Find all current US senators",
  schema = c(name = "character", state = "character", party = "character")
)
audit <- asa_audit(senators, backend = "claude_code")
print(audit)

# Audit with known universe for precise completeness check
audit <- asa_audit(senators, known_universe = state.abb)

# Interactive mode for complex audits
asa_audit(senators, backend = "claude_code", interactive = TRUE)

# Use LangGraph backend
audit <- asa_audit(senators, backend = "langgraph", agent = agent)

## End(Not run)
```

`asa_audit_result`      *Constructor for `asa_audit_result` Objects*

**Description**

Creates an S3 object representing the result of a data quality audit.

**Usage**

```
asa_audit_result(
  data,
  audit_summary,
  issues,
  recommendations,
  completeness_score,
```

```

  consistency_score,
  backend_used,
  elapsed_time,
  query = NULL,
  checks = NULL
)

```

### Arguments

<code>data</code>	data.frame with original data plus audit columns (_audit_flag, _audit_notes)
<code>audit_summary</code>	Character string with high-level findings
<code>issues</code>	List of identified issues with severity and descriptions
<code>recommendations</code>	Character vector of suggested remediation queries
<code>completeness_score</code>	Numeric 0-1 score for data completeness
<code>consistency_score</code>	Numeric 0-1 score for data consistency
<code>backend_used</code>	Which backend performed the audit ("claude_code" or "langgraph")
<code>elapsed_time</code>	Execution time in seconds
<code>query</code>	The original query (if available)
<code>checks</code>	Character vector of checks that were performed

### Value

An object of class `asa_audit_result`

## `asa_config`

*Create ASA Configuration Object*

### Description

Creates a configuration object that encapsulates all settings for ASA tasks. This provides a unified way to configure backend, model, search, temporal, and resource settings in a single object.

### Usage

```

asa_config(
  backend = NULL,
  model = NULL,
  conda_env = NULL,
  proxy = NULL,
  workers = NULL,
  timeout = NULL,
  rate_limit = NULL,
  memory_folding = NULL,
  memory_threshold = NULL,
  memory_keep_recent = NULL,
  temporal = NULL,
  search = NULL
)

```

## Arguments

backend	LLM backend: "openai", "groq", "xai", "exo", "openrouter"
model	Model identifier (e.g., "gpt-4.1-mini")
conda_env	Conda environment name (default: "asa_env")
proxy	SOCKS5 proxy URL or NULL to disable
workers	Number of parallel workers for batch operations
timeout	Request timeout in seconds
rate_limit	Requests per second
memory_folding	Enable DeepAgent-style memory folding
memory_threshold	Messages before folding triggers
memory_keep_recent	Messages to preserve after folding
temporal	Temporal filtering options (use <code>temporal_options()</code> )
search	Search configuration (use <code>search_options()</code> )

## Details

The configuration object can be passed to `run_task()`, `run_task_batch()`, `asa_enumerate()`, and other functions to provide consistent settings across operations.

## Value

An object of class `asa_config`

## See Also

[temporal\\_options](#), [search\\_options](#)

## Examples

```
## Not run:  
# Create configuration  
config <- asa_config(  
  backend = "openai",  
  model = "gpt-4.1-mini",  
  workers = 4,  
  temporal = temporal_options(time_filter = "y")  
)  
  
# Use with run_task  
result <- run_task(prompt, config = config)  
## End(Not run)
```

---

ASA\_DEFAULT\_BACKEND      *Default Backend*

---

**Description**

Default Backend

**Usage**

ASA\_DEFAULT\_BACKEND

**Format**

An object of class `character` of length 1.

---

---

ASA\_DEFAULT\_BUDGET\_QUERIES  
Default Budget: *Queries*

---

**Description**

Default Budget: Queries

**Usage**

ASA\_DEFAULT\_BUDGET\_QUERIES

**Format**

An object of class `integer` of length 1.

---

---

ASA\_DEFAULT\_BUDGET\_TIME  
Default Budget: *Time (seconds)*

---

**Description**

Default Budget: Time (seconds)

**Usage**

ASA\_DEFAULT\_BUDGET\_TIME

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_BUDGET\_TOKENS

*Default Budget: Tokens*

---

**Description**

Default Budget: Tokens

**Usage**

ASA\_DEFAULT\_BUDGET\_TOKENS

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_CONDA\_ENV *Default Conda Environment*

---

**Description**

Default Conda Environment

**Usage**

ASA\_DEFAULT\_CONDA\_ENV

**Format**

An object of class `character` of length 1.

---

ASA\_DEFAULT\_INTER\_SEARCH\_DELAY

*Default Inter-Search Delay (seconds)*

---

**Description**

Default Inter-Search Delay (seconds)

**Usage**

ASA\_DEFAULT\_INTER\_SEARCH\_DELAY

**Format**

An object of class `numeric` of length 1.

---

ASA\_DEFAULT\_MAX\_RESULTS

*Default Max Search Results*

---

**Description**

Default Max Search Results

**Usage**

ASA\_DEFAULT\_MAX\_RESULTS

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_MAX\_RETRIES

*Default Max Retries*

---

**Description**

Default Max Retries

**Usage**

ASA\_DEFAULT\_MAX\_RETRIES

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_MAX\_ROUNDS

*Default Max Rounds for Enumeration*

---

**Description**

Default Max Rounds for Enumeration

**Usage**

ASA\_DEFAULT\_MAX\_ROUNDS

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_MEMORY\_FOLDING

*Default Memory Folding Enabled*

---

**Description**

Default Memory Folding Enabled

**Usage**

ASA\_DEFAULT\_MEMORY\_FOLDING

**Format**

An object of class logical of length 1.

---

ASA\_DEFAULT\_MEMORY\_KEEP\_RECENT

*Default Messages to Keep After Folding*

---

**Description**

Default Messages to Keep After Folding

**Usage**

ASA\_DEFAULT\_MEMORY\_KEEP\_RECENT

**Format**

An object of class integer of length 1.

---

ASA\_DEFAULT\_MEMORY\_THRESHOLD

*Default Memory Threshold (messages before folding)*

---

**Description**

Default Memory Threshold (messages before folding)

**Usage**

ASA\_DEFAULT\_MEMORY\_THRESHOLD

**Format**

An object of class integer of length 1.

---

ASA_DEFAULT_MODEL	<i>Default Model</i>
-------------------	----------------------

---

**Description**

Default Model

**Usage**

ASA\_DEFAULT\_MODEL

**Format**

An object of class `character` of length 1.

---

ASA_DEFAULT_NOVELTY_MIN	<i>Default Minimum Novelty Rate</i>
-------------------------	-------------------------------------

---

**Description**

Default Minimum Novelty Rate

**Usage**

ASA\_DEFAULT\_NOVELTY\_MIN

**Format**

An object of class `numeric` of length 1.

---

ASA_DEFAULT_NOVELTY_WINDOW	<i>Default Novelty Window</i>
----------------------------	-------------------------------

---

**Description**

Default Novelty Window

**Usage**

ASA\_DEFAULT\_NOVELTY\_WINDOW

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_PAGE\_LOAD\_WAIT

*Default Page Load Wait (seconds)*

---

**Description**

Default Page Load Wait (seconds)

**Usage**

ASA\_DEFAULT\_PAGE\_LOAD\_WAIT

**Format**

An object of class numeric of length 1.

---

ASA\_DEFAULT\_PLATEAU\_ROUNDS

*Default Plateau Rounds for Stopping*

---

**Description**

Default Plateau Rounds for Stopping

**Usage**

ASA\_DEFAULT\_PLATEAU\_ROUNDS

**Format**

An object of class integer of length 1.

---

ASA\_DEFAULT\_PROXY

*Default Proxy URL (Tor SOCKS5)*

---

**Description**

Default Proxy URL (Tor SOCKS5)

**Usage**

ASA\_DEFAULT\_PROXY

**Format**

An object of class character of length 1.

---

ASA\_DEFAULT\_RATE\_LIMIT

*Default Rate Limit (requests per second)*

---

**Description**

Default Rate Limit (requests per second)

**Usage**

```
ASA_DEFAULT_RATE_LIMIT
```

**Format**

An object of class numeric of length 1.

---

---

ASA\_DEFAULT\_TEMPERATURES

*Default Temperatures by Backend*

---

**Description**

Default Temperatures by Backend

**Usage**

```
ASA_DEFAULT_TEMPERATURES
```

**Format**

An object of class list of length 5.

---

---

ASA\_DEFAULT\_TIMEOUT

*Default Request Timeout (seconds)*

---

**Description**

Default Request Timeout (seconds)

**Usage**

```
ASA_DEFAULT_TIMEOUT
```

**Format**

An object of class integer of length 1.

---

ASA\_DEFAULT\_WIKI\_CHARS

*Default Wikipedia Content Chars*

---

**Description**

Default Wikipedia Content Chars

**Usage**

ASA\_DEFAULT\_WIKI\_CHARS

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_WIKI\_TOP\_K

*Default Wikipedia Top K Results*

---

**Description**

Default Wikipedia Top K Results

**Usage**

ASA\_DEFAULT\_WIKI\_TOP\_K

**Format**

An object of class `integer` of length 1.

---

ASA\_DEFAULT\_WORKERS

*Default Max Workers for Enumeration*

---

**Description**

Default Max Workers for Enumeration

**Usage**

ASA\_DEFAULT\_WORKERS

**Format**

An object of class `integer` of length 1.

## Description

Performs intelligent open-ended research tasks using multi-agent orchestration. Decomposes complex queries into sub-tasks, executes parallel searches, and aggregates results into structured output (data.frame, CSV, or JSON).

## Usage

```
asa_enumerate(
  query,
  schema = NULL,
  output = c("data.frame", "csv", "json"),
  workers = NULL,
  max_rounds = NULL,
  budget = list(queries = 50L, tokens = 200000L, time_sec = 300L),
  stop_policy = list(target_items = NULL, plateau_rounds = 2L, novelty_min = 0.05,
    novelty_window = 20L),
  sources = list(web = TRUE, wikipedia = TRUE, wikidata = TRUE),
  temporal = NULL,
  pagination = TRUE,
  progress = TRUE,
  include_provenance = FALSE,
  checkpoint = TRUE,
  checkpoint_dir = tempdir(),
  resume_from = NULL,
  agent = NULL,
  backend = NULL,
  model = NULL,
  conda_env = NULL,
  verbose = TRUE
)
```

## Arguments

<code>query</code>	Character string describing the research goal. Examples: "Find all current US senators with their state, party, and term end date"
<code>schema</code>	Named character vector defining the output schema. Names are column names, values are R types ("character", "numeric", "logical"). Use <code>NULL</code> or <code>"auto"</code> for LLM-proposed schema.
<code>output</code>	Output format: <code>"data.frame"</code> (default), <code>"csv"</code> , or <code>"json"</code> .
<code>workers</code>	Number of parallel search workers. Defaults to value from <code>ASA_DEFAULT_WORKERS</code> (typically 4).
<code>max_rounds</code>	Maximum research iterations. Defaults to value from <code>ASA_DEFAULT_MAX_ROUNDS</code> (typically 8).
<code>budget</code>	Named list with resource limits: <ul style="list-style-type: none"> <li>• <code>queries</code>: Maximum search queries (default: 50)</li> </ul>

	<ul style="list-style-type: none"> <li>• tokens: Maximum LLM tokens (default: 200000)</li> <li>• time_sec: Maximum execution time in seconds (default: 300)</li> </ul>
stop_policy	Named list with stopping criteria: <ul style="list-style-type: none"> <li>• target_items: Stop when this many items found (NULL = unknown)</li> <li>• plateau_rounds: Stop after N rounds with no new items (default: 2)</li> <li>• novelty_min: Minimum new items ratio per round (default: 0.05)</li> <li>• novelty_window: Window size for novelty calculation (default: 20)</li> </ul>
sources	Named list controlling which sources to use: <ul style="list-style-type: none"> <li>• web: Use DuckDuckGo web search (default: TRUE)</li> <li>• wikipedia: Use Wikipedia (default: TRUE)</li> <li>• wikidata: Use Wikidata SPARQL for authoritative enumerations (default: TRUE)</li> </ul>
temporal	Named list for temporal filtering: <ul style="list-style-type: none"> <li>• after: ISO 8601 date string (e.g., "2020-01-01") - results after this date</li> <li>• before: ISO 8601 date string (e.g., "2024-01-01") - results before this date</li> <li>• time_filter: DuckDuckGo time filter ("d", "w", "m", "y") for day/week/month/year</li> <li>• strictness: "best_effort" (default) or "strict" (verifies dates via metadata)</li> <li>• use_wayback: Use Wayback Machine for strict pre-date guarantees (default: FALSE)</li> </ul>
pagination	Enable pagination for large result sets (default: TRUE).
progress	Show progress bar and status updates (default: TRUE).
include_provenance	Include source URLs and confidence per row (default: FALSE).
checkpoint	Enable auto-save after each round (default: TRUE).
checkpoint_dir	Directory for checkpoint files (default: tempdir()).
resume_from	Path to checkpoint file to resume from (default: NULL).
agent	An initialized <code>asa_agent</code> object. If NULL, uses the current agent or creates a new one with specified backend/model.
backend	LLM backend if creating new agent: "openai", "groq", "xai", "openrouter".
model	Model identifier if creating new agent.
conda_env	Conda environment name (default: "asa_env").
verbose	Print status messages (default: TRUE).

## Details

The function uses a multi-agent architecture:

1. **Planner:** Decomposes query into facets and identifies authoritative sources
2. **Dispatcher:** Spawns parallel workers for each facet
3. **Workers:** Execute searches using DDG, Wikipedia, and Wikidata
4. **Extractor:** Normalizes results to match schema
5. **Deduper:** Removes duplicates using hash + fuzzy matching
6. **Stopper:** Evaluates stopping criteria (novelty, budget, saturation)

For known entity types (US senators, countries, Fortune 500), Wikidata provides authoritative enumerations with complete, verified data.

## Value

An object of class `asa_enumerate_result` containing:

- `data`: `data.frame` with results matching the schema
- `status`: "complete", "partial", or "failed"
- `stop_reason`: Why the search stopped
- `metrics`: List with rounds, queries\_used, novelty\_curve, coverage
- `provenance`: If `include_provenance=TRUE`, source info per row
- `checkpoint_file`: Path to checkpoint if saved

## Checkpointing

With `checkpoint=TRUE`, state is saved after each round. If interrupted, use `resume_from` to continue from the last checkpoint:

```
result <- asa_enumerate(query, resume_from = "/path/to/checkpoint.rds")
```

## Schema

The schema defines expected output columns:

```
schema = c(name = "character", state = "character", party = "character")
```

With `schema = "auto"`, the planner agent proposes a schema based on the query.

## See Also

[run\\_task](#), [initialize\\_agent](#)

## Examples

```
## Not run:
# Find all US senators
senators <- asa_enumerate(
  query = "Find all current US senators with state, party, and term end date",
  schema = c(name = "character", state = "character",
             party = "character", term_end = "character"),
  stop_policy = list(target_items = 100),
  include_provenance = TRUE
)
head(senators$data)

# Find countries with auto schema
countries <- asa_enumerate(
  query = "Find all countries with their capitals and populations",
  schema = "auto",
  output = "csv"
)

# Resume from checkpoint
result <- asa_enumerate(
  query = "Find Fortune 500 CEOs",
  resume_from = "/tmp/asa_enumerate_abc123.rds"
)
```

```

# Temporal filtering: results from specific date range
companies_2020s <- asa_enumerate(
  query = "Find tech companies founded recently",
  temporal = list(
    after = "2020-01-01",
    before = "2024-01-01",
    strictness = "best_effort"
  )
)

# Temporal filtering: past year with DuckDuckGo time filter
recent_news <- asa_enumerate(
  query = "Find AI research breakthroughs",
  temporal = list(
    time_filter = "y" # past year
  )
)

# Strict temporal filtering with Wayback Machine
historical <- asa_enumerate(
  query = "Find Fortune 500 companies",
  temporal = list(
    before = "2015-01-01",
    strictness = "strict",
    use_wayback = TRUE
  )
)

## End(Not run)

```

**asa\_enumerate\_result** *Constructor for asa\_enumerate\_result Objects*

## Description

Creates an S3 object representing the result of an enumeration task.

## Usage

```

asa_enumerate_result(
  data,
  status,
  stop_reason,
  metrics,
  provenance = NULL,
  plan = NULL,
  checkpoint_file = NULL,
  query = NULL,
  schema = NULL
)

```

**Arguments**

data	data.frame containing the enumeration results
status	Result status: "complete", "partial", or "failed"
stop_reason	Why the enumeration stopped (e.g., "target_reached", "novelty_plateau")
metrics	List with execution metrics (rounds, queries_used, etc.)
provenance	Optional data.frame with source information per row
plan	The enumeration plan from the planner agent
checkpoint_file	Path to saved checkpoint file
query	The original enumeration query
schema	The schema used for extraction

**Value**

An object of class `asa_enumerate_result`

ASA\_OUTPUT\_FORMATS      *Valid Output Formats*

**Description**

Valid Output Formats

**Usage**

`ASA_OUTPUT_FORMATS`

**Format**

An object of class `character` of length 3.

ASA\_PRINT\_WIDTH      *Print Width for Output*

**Description**

Print Width for Output

**Usage**

`ASA_PRINT_WIDTH`

**Format**

An object of class `integer` of length 1.

---

ASA\_RATE\_LIMIT\_WAIT      *Rate Limit Wait Time (seconds)*

---

**Description**

Rate Limit Wait Time (seconds)

**Usage**

ASA\_RATE\_LIMIT\_WAIT

**Format**

An object of class `integer` of length 1.

---

ASA\_RECUSION\_LIMIT\_FOLDING  
Recursion Limit with Memory Folding

---

**Description**

Recursion Limit with Memory Folding

**Usage**

ASA\_RECUSION\_LIMIT\_FOLDING

**Format**

An object of class `integer` of length 1.

---

ASA\_RECUSION\_LIMIT\_STANDARD  
Recursion Limit without Memory Folding

---

**Description**

Recursion Limit without Memory Folding

**Usage**

ASA\_RECUSION\_LIMIT\_STANDARD

**Format**

An object of class `integer` of length 1.

asa_response	<i>Constructor for asa_response Objects</i>
--------------	---

## Description

Creates an S3 object representing an agent response.

## Usage

```
asa_response(
    message,
    status_code,
    raw_response,
    trace,
    elapsed_time,
    fold_count,
    prompt
)
```

## Arguments

<code>message</code>	The final response text
<code>status_code</code>	Status code (200 = success, 100 = error)
<code>raw_response</code>	The full Python response object
<code>trace</code>	Full text trace of agent execution
<code>elapsed_time</code>	Execution time in minutes
<code>fold_count</code>	Number of memory folds performed
<code>prompt</code>	The original prompt

## Value

An object of class `asa_response`

asa_result	<i>Constructor for asa_result Objects</i>
------------	---

## Description

Creates an S3 object representing the result of a research task.

## Usage

```
asa_result(prompt, message, parsed, raw_output, elapsed_time, status)
```

**Arguments**

prompt	The original prompt
message	The agent's response text
parsed	Parsed output (list or NULL)
raw_output	Full agent trace
elapsed_time	Execution time in minutes
status	Status ("success" or "error")

**Value**

An object of class `asa_result`

---

ASA\_STATUS\_ERROR      *Status Code: Error*

---

**Description**

Status Code: Error

**Usage**

`ASA_STATUS_ERROR`

**Format**

An object of class `integer` of length 1.

---

ASA\_STATUS\_SUCCESS      *Status Code: Success*

---

**Description**

Status Code: Success

**Usage**

`ASA_STATUS_SUCCESS`

**Format**

An object of class `integer` of length 1.

---

ASA\_SUPPORTED\_BACKENDS

*Supported Backends*

---

**Description**

Supported Backends

**Usage**

ASA\_SUPPORTED\_BACKENDS

**Format**

An object of class character of length 5.

---

## ASA\_TEMPORAL\_STRICTNESS

*Valid Temporal Strictness Levels*

---

**Description**

Valid Temporal Strictness Levels

**Usage**

ASA\_TEMPORAL\_STRICTNESS

**Format**

An object of class character of length 2.

---

## ASA\_TIME\_FILTERS

*Valid Temporal Time Filters*

---

**Description**

Valid Temporal Time Filters

**Usage**

ASA\_TIME\_FILTERS

**Format**

An object of class character of length 4.

---

ASA_TRUNCATE_LENGTH	<i>String Truncation Length</i>
---------------------	---------------------------------

---

**Description**

String Truncation Length

**Usage**

```
ASA_TRUNCATE_LENGTH
```

**Format**

An object of class `integer` of length 1.

---

build_backend	<i>Build the Python Backend Environment</i>
---------------	---

---

**Description**

Creates a conda environment with all required Python dependencies for the asa search agent, including LangChain, LangGraph, and search tools.

**Usage**

```
build_backend(conda_env = "asa_env", conda = "auto", python_version = "3.13")
```

**Arguments**

conda_env	Name of the conda environment (default: "asa_env")
conda	Path to conda executable (default: "auto")
python_version	Python version to use (default: "3.13")

**Details**

This function creates a new conda environment and installs the following Python packages:

- langchain\_groq, langchain\_community, langchain\_openai
- langgraph
- ddgs (DuckDuckGo search)
- selenium, primp (browser automation)
- beautifulsoup4, requests
- fake\_headers, httpx
- pysocks, socksio (proxy support)

**Value**

Invisibly returns NULL; called for side effects.

## Examples

```
## Not run:
# Create the default environment
build_backend()

# Create with a custom name
build_backend(conda_env = "my_asa_env")

## End(Not run)
```

### **build\_prompt**

*Build a Task Prompt from Template*

## Description

Creates a formatted prompt by substituting variables into a template.

## Usage

```
build_prompt(template, ...)
```

## Arguments

template	A character string with placeholders in the form {variable_name}
...	Named arguments to substitute into the template

## Value

A formatted prompt string

## Examples

```
## Not run:
prompt <- build_prompt(
  template = "Find information about {{name}} in {{country}} during {{year}}",
  name = "Marie Curie",
  country = "France",
  year = 1903
)
## End(Not run)
```

---

check_backend	<i>Check Python Environment Availability</i>
---------------	--

---

### Description

Checks if the required Python environment and packages are available.

### Usage

```
check_backend(conda_env = "asa_env")
```

### Arguments

conda\_env      Name of the conda environment to check

### Value

A list with components:

- available: Logical, TRUE if environment is ready
- conda\_env: Name of the environment checked
- python\_version: Python version if available
- missing\_packages: Character vector of missing packages (if any)

### Examples

```
## Not run:  
status <- check_backend()  
if (!status$available) {  
  build_backend()  
}  
  
## End(Not run)
```

---

clean_whitespace	<i>Clean Whitespace</i>
------------------	-------------------------

---

### Description

Normalizes whitespace in a string by collapsing multiple spaces and trimming leading/trailing whitespace.

### Usage

```
clean_whitespace(x)
```

### Arguments

x      Character string

**Value**

Cleaned string

---



---

configure\_search

*Configure Python Search Parameters*

---

**Description**

Sets global configuration values for the Python search module. These values control timeouts, retry behavior, and result limits.

**Usage**

```
configure_search(
    max_results = NULL,
    timeout = NULL,
    max_retries = NULL,
    retry_delay = NULL,
    backoff_multiplier = NULL,
    captcha_backoff_base = NULL,
    page_load_wait = NULL,
    inter_search_delay = NULL,
    conda_env = "asa_env"
)
```

**Arguments**

max_results	Maximum number of search results to return (default: 10)
timeout	HTTP request timeout in seconds (default: 15)
max_retries	Maximum retry attempts on failure (default: 3)
retry_delay	Initial delay between retries in seconds (default: 2)
backoff_multiplier	Multiplier for exponential backoff (default: 1.5)
captcha_backoff_base	Base multiplier for CAPTCHA backoff (default: 3)
page_load_wait	Wait time after page load in seconds (default: 2)
inter_search_delay	Delay between consecutive searches in seconds (default: 0.5)
conda_env	Name of the conda environment (default: "asa_env")

**Value**

Invisibly returns a list with the current configuration

## Examples

```
## Not run:  
# Increase timeout for slow connections  
configure_search(timeout = 30, max_retries = 5)  
  
# Get more results  
configure_search(max_results = 20)  
  
# Add delay between searches to avoid rate limiting  
configure_search(inter_search_delay = 2.0)  
  
## End(Not run)
```

---

## configure\_search\_logging

*Configure Python Search Logging Level*

---

## Description

Sets the logging level for the Python search module. This controls how much diagnostic output is produced during web searches.

## Usage

```
configure_search_logging(level = "WARNING", conda_env = "asa_env")
```

## Arguments

level	Log level: "DEBUG", "INFO", "WARNING" (default), "ERROR", or "CRITICAL"
conda_env	Name of the conda environment (default: "asa_env")

## Details

Log levels from most to least verbose:

- DEBUG: Detailed diagnostic information for debugging
- INFO: General operational information
- WARNING: Indicates something unexpected but not an error (default)
- ERROR: Serious problems that prevented an operation
- CRITICAL: Very serious errors

## Value

Invisibly returns the current logging level

## Examples

```

## Not run:
# Enable verbose debugging output
configure_search_logging("DEBUG")

# Run a search (will show detailed logs)
result <- run_task("What is the population of Tokyo?", agent = agent)

# Disable verbose output
configure_search_logging("WARNING")

## End(Not run)

```

**configure\_temporal**     *Configure Temporal Filtering for Search*

## Description

Sets or clears temporal filtering on the DuckDuckGo search tool. This affects all subsequent searches until changed or cleared.

## Usage

```
configure_temporal(time_filter = NULL)
```

## Arguments

<code>time_filter</code>	DuckDuckGo time filter: "d" (day), "w" (week), "m" (month), "y" (year), or NULL/NA/"none" to clear
--------------------------	---

## Details

This function modifies the search tool's time parameter, which is passed to DuckDuckGo as the `df` parameter. The filter restricts results to content indexed within the specified time period.

Note: This only affects DuckDuckGo searches. For Wikidata queries with temporal filtering, use `asa_enumerate()` with its `temporal` parameter.

## Value

Invisibly returns the previous time filter setting

## Time Filter Values

- "d": Past 24 hours (day)
- "w": Past 7 days (week)
- "m": Past 30 days (month)
- "y": Past 365 days (year)
- NULL, NA, or "none": No time restriction (default)

**See Also**[run\\_task](#), [asa\\_enumerate](#)**Examples**

```
## Not run:  
# Restrict to past year  
configure_temporal("y")  
result <- run_task("Find recent AI breakthroughs", agent = agent)  
  
# Clear temporal filter  
configure_temporal(NULL)  
  
# Past week only  
configure_temporal("w")  
  
## End(Not run)
```

---

**decode\_html***Decode HTML Entities*

---

**Description**

Converts HTML entities to their character equivalents.

**Usage**

```
decode_html(x)
```

**Arguments**

x	Character string with HTML entities
---	-------------------------------------

**Value**

Decoded string

---

**extract\_agent\_results** *Extract Structured Data from Agent Traces*

---

**Description**

Parses raw agent output to extract search snippets, Wikipedia content, URLs, JSON data, and search tier information. This is the main function for post-processing agent traces.

**Usage**

```
extract_agent_results(raw_output)
```

**Arguments**

raw_output	Raw output string from agent invocation (the trace field from an asa_response object)
------------	---

**Value**

A list with components:

- search\_snippets: Character vector of search result content
- search\_urls: Character vector of URLs from search results
- wikipedia\_snippets: Character vector of Wikipedia content
- json\_data: Extracted JSON data as a list (if present)
- search\_tiers: Character vector of unique search tiers used (e.g., "primp", "selenium", "ddgs", "requests")

**Examples**

```
## Not run:
response <- run_agent("Who is the president of France?", agent)
extracted <- extract_agent_results(response$trace)
print(extracted$search_snippets)
print(extracted$search_tiers) # Shows which search tier was used

## End(Not run)
```

**extract\_search\_snippets**

*Extract Search Snippets by Source Number*

**Description**

Extracts content from Search tool messages in the agent trace.

**Usage**

```
extract_search_snippets(text)
```

**Arguments**

text	Raw agent trace text
------	----------------------

**Value**

Character vector of search snippets, ordered by source number

**Examples**

```
## Not run:
snippets <- extract_search_snippets(response$trace)

## End(Not run)
```

---

extract\_search\_tiers    *Extract Search Tier Information*

---

### Description

Extracts which search tier was used from the agent trace. The search module uses a multi-tier fallback system:

- primp: Fast HTTP client with browser impersonation (Tier 0)
- selenium: Headless browser for JS-rendered content (Tier 1)
- ddgs: Standard DDGS Python library (Tier 2)
- requests: Raw POST to DuckDuckGo HTML endpoint (Tier 3)

### Usage

```
extract_search_tiers(text)
```

### Arguments

text              Raw agent trace text

### Value

Character vector of unique tier names encountered (e.g., "primp", "selenium", "ddgs", "requests")

### Examples

```
## Not run:  
tiers <- extract_search_tiers(response$trace)  
print(tiers) # e.g., "primp"  
  
## End(Not run)
```

---

extract\_urls              *Extract URLs by Source Number*

---

### Description

Extracts URLs from Search tool messages in the agent trace.

### Usage

```
extract_urls(text)
```

### Arguments

text              Raw agent trace text

**Value**

Character vector of URLs, ordered by source number

**Examples**

```
## Not run:
urls <- extract_urls(response$trace)

## End(Not run)
```

---

**extract\_wikipedia\_content**

*Extract Wikipedia Content*

---

**Description**

Extracts content from Wikipedia tool messages in the agent trace.

**Usage**

```
extract_wikipedia_content(text)
```

**Arguments**

text	Raw agent trace text
------	----------------------

**Value**

Character vector of Wikipedia snippets

**Examples**

```
## Not run:
wiki <- extract_wikipedia_content(response$trace)

## End(Not run)
```

---

**format\_duration**

*Format Time Duration*

---

**Description**

Formats a numeric duration (in minutes) as a human-readable string.

**Usage**

```
format_duration(minutes)
```

**Arguments**

minutes      Numeric duration in minutes

**Value**

Formatted string

---

get\_agent      *Get the Current Agent*

---

**Description**

Returns the currently initialized agent, or NULL if not initialized.

**Usage**

get\_agent()

**Value**

An asa\_agent object or NULL

**Examples**

```
## Not run:  
agent <- get_agent()  
if (is.null(agent)) {  
  agent <- initialize_agent()  
}  
  
## End(Not run)
```

---

get\_tor\_ip      *Get External IP via Tor*

---

**Description**

Retrieves the external IP address as seen through Tor proxy.

**Usage**

get\_tor\_ip(proxy = "socks5h://127.0.0.1:9050")

**Arguments**

proxy      Tor proxy URL

**Value**

IP address string or NA on failure

## Examples

```
## Not run:
ip <- get_tor_ip()
message("Current Tor IP: ", ip)

## End(Not run)
```

`initialize_agent`      *Initialize the ASA Search Agent*

## Description

Initializes the Python environment and creates the LangGraph agent with search tools (Wikipedia, DuckDuckGo). The agent can use multiple LLM backends and supports DeepAgent-style memory folding.

## Usage

```
initialize_agent(
  backend = "openai",
  model = "gpt-4.1-mini",
  conda_env = "asa_env",
  proxy = "socks5h://127.0.0.1:9050",
  use_memory_folding = TRUE,
  memory_threshold = 4L,
  memory_keep_recent = 2L,
  rate_limit = 0.2,
  timeout = 120L,
  verbose = TRUE
)
```

## Arguments

<code>backend</code>	LLM backend to use. One of: "openai", "groq", "xai", "exo", "openrouter"
<code>model</code>	Model identifier (e.g., "gpt-4.1-mini", "llama-3.3-70b-versatile")
<code>conda_env</code>	Name of the conda environment with Python dependencies
<code>proxy</code>	SOCKS5 proxy URL for Tor (default: "socks5h://127.0.0.1:9050"). Set to NULL to disable proxy.
<code>use_memory_folding</code>	Enable DeepAgent-style memory compression (default: TRUE)
<code>memory_threshold</code>	Number of messages before folding triggers (default: 4)
<code>memory_keep_recent</code>	Number of recent messages to preserve after folding (default: 2)
<code>rate_limit</code>	Requests per second for rate limiting (default: 0.2)
<code>timeout</code>	Request timeout in seconds (default: 120)
<code>verbose</code>	Print status messages (default: TRUE)

## Details

The agent is created with two tools:

- Wikipedia: For looking up encyclopedic information
- DuckDuckGo Search: For web searches with a 4-tier fallback system (PRIMP -> Selenium -> DDGS library -> raw requests)

Memory folding (enabled by default) compresses older messages into a summary to manage context length in long conversations, following the DeepAgent paper.

## Value

An object of class `asa_agent` containing the initialized agent and configuration.

## API Keys

The following environment variables should be set based on your backend:

- OpenAI: OPENAI\_API\_KEY
- Groq: GROQ\_API\_KEY
- xAI: XAI\_API\_KEY
- OpenRouter: OPENROUTER\_API\_KEY

## OpenRouter Models

When using the "openrouter" backend, model names must be in provider/model-name format.  
Examples:

- "openai/gpt-4o"
- "anthropic/clause-3-sonnet"
- "google/gemma-2-9b-it:free"
- "meta-llama/llama-3-70b-instruct"

See <https://openrouter.ai/models> for available models.

## See Also

[run\\_task](#), [run\\_task\\_batch](#)

## Examples

```
## Not run:  
# Initialize with OpenAI  
agent <- initialize_agent(  
  backend = "openai",  
  model = "gpt-4.1-mini"  
)  
  
# Initialize with Groq and custom settings  
agent <- initialize_agent(  
  backend = "groq",  
  model = "llama-3.3-70b-versatile",  
  use_memory_folding = FALSE,  
  proxy = NULL # No Tor proxy
```

```
)
# Initialize with OpenRouter (access to 100+ models)
agent <- initialize_agent(
  backend = "openrouter",
  model = "anthropic/clause-3-sonnet" # Note: provider/model format
)
## End(Not run)
```

**is\_tor\_running**      *Check if Tor is Running*

### Description

Checks if Tor is running and accessible on the default port.

### Usage

```
is_tor_running(port = 9050L)
```

### Arguments

port	Port number (default: 9050)
------	-----------------------------

### Value

Logical indicating if Tor appears to be running

### Examples

```
## Not run:
if (!is_tor_running()) {
  message("Start Tor with: brew services start tor")
}

## End(Not run)
```

**json\_escape**      *Clean Text for JSON Output*

### Description

Escapes special characters in text for safe inclusion in JSON strings.

### Usage

```
json_escape(x)
```

**Arguments**

x Character string to escape

**Value**

Escaped string

---

print.asa\_agent

*Print Method for asa\_agent Objects*

---

**Description**

Print Method for asa\_agent Objects

**Usage**

```
## S3 method for class 'asa_agent'  
print(x, ...)
```

**Arguments**

x An asa\_agent object  
... Additional arguments (ignored)

**Value**

Invisibly returns the object

---

print.asa\_audit\_result

*Print Method for asa\_audit\_result Objects*

---

**Description**

Print Method for asa\_audit\_result Objects

**Usage**

```
## S3 method for class 'asa_audit_result'  
print(x, n = 6, ...)
```

**Arguments**

x An asa\_audit\_result object  
n Number of data rows to preview (default: 6)  
... Additional arguments (ignored)

**Value**

Invisibly returns the object

---

**print.asa\_config**      *Print Method for asa\_config Objects*

---

### Description

Print Method for asa\_config Objects

### Usage

```
## S3 method for class 'asa_config'  
print(x, ...)
```

### Arguments

x	An asa_config object
...	Additional arguments (ignored)

### Value

Invisibly returns the object

---

**print.asa\_enumerate\_result**      *Print Method for asa\_enumerate\_result Objects*

---

### Description

Print Method for asa\_enumerate\_result Objects

### Usage

```
## S3 method for class 'asa_enumerate_result'  
print(x, n = 6, ...)
```

### Arguments

x	An asa_enumerate_result object
n	Number of data rows to preview (default: 6)
...	Additional arguments (ignored)

### Value

Invisibly returns the object

---

print.asa\_response      *Print Method for asa\_response Objects*

---

## Description

Print Method for asa\_response Objects

## Usage

```
## S3 method for class 'asa_response'  
print(x, ...)
```

## Arguments

x	An asa_response object
...	Additional arguments (ignored)

## Value

Invisibly returns the object

---

print.asa\_result      *Print Method for asa\_result Objects*

---

## Description

Print Method for asa\_result Objects

## Usage

```
## S3 method for class 'asa_result'  
print(x, ...)
```

## Arguments

x	An asa_result object
...	Additional arguments (ignored)

## Value

Invisibly returns the object

---

**print.asa\_search***Print Method for asa\_search Objects*

---

**Description**

Print Method for asa\_search Objects

**Usage**

```
## S3 method for class 'asa_search'  
print(x, ...)
```

**Arguments**

x	An asa_search object
...	Additional arguments (ignored)

---

**print.asa\_temporal***Print Method for asa\_temporal Objects*

---

**Description**

Print Method for asa\_temporal Objects

**Usage**

```
## S3 method for class 'asa_temporal'  
print(x, ...)
```

**Arguments**

x	An asa_temporal object
...	Additional arguments (ignored)

**Value**

Invisibly returns the object

---

print2

*Print Utility*

---

### Description

Wrapper around cat for consistent output formatting.

### Usage

```
print2(...)
```

### Arguments

...	Arguments passed to cat
-----	-------------------------

---

process\_outputs

*Process Multiple Agent Outputs*

---

### Description

Processes a data frame of raw agent outputs, extracting structured data.

### Usage

```
process_outputs(df, parallel = FALSE, workers = 10L)
```

### Arguments

df	Data frame with a 'raw_output' column containing agent traces
parallel	Use parallel processing
workers	Number of workers

### Value

The input data frame with additional extracted columns: search\_count, wiki\_count, and any JSON fields found

reset\_agent

*Reset the Agent***Description**

Clears the initialized agent state, forcing reinitialization on next use. Also closes any open HTTP clients to prevent resource leaks.

**Usage**

```
reset_agent()
```

**Value**

Invisibly returns NULL

rotate\_tor\_circuit

*Rotate Tor Circuit***Description**

Requests a new Tor circuit by restarting the Tor service.

**Usage**

```
rotate_tor_circuit(method = c("brew", "systemctl", "signal"), wait = 12L)
```

**Arguments**

method            Method to restart: "brew" (macOS), "systemctl" (Linux), or "signal"

wait            Seconds to wait for new circuit (default: 12)

**Value**

Invisibly returns NULL

**Examples**

```
## Not run:  
rotate_tor_circuit()  
  
## End(Not run)
```

---

run_task	<i>Run a Structured Task with the Agent</i>
----------	---

---

## Description

Executes a research task using the AI search agent with a structured prompt and returns parsed results. This is the primary function for running agent tasks.

## Usage

```
run_task(  
  prompt,  
  output_format = "text",  
  temporal = NULL,  
  config = NULL,  
  agent = NULL,  
  verbose = FALSE  
)
```

## Arguments

<code>prompt</code>	The task prompt or question for the agent to research
<code>output_format</code>	Expected output format. One of: <ul style="list-style-type: none"><li>• "text": Returns response text (default)</li><li>• "json": Parse response as JSON</li><li>• "raw": Include full trace in result for debugging</li><li>• Character vector: Extract specific fields from response</li></ul>
<code>temporal</code>	Named list or <code>asa_temporal</code> object for temporal filtering: <ul style="list-style-type: none"><li>• <code>time_filter</code>: DuckDuckGo time filter - "d" (day), "w" (week), "m" (month), "y" (year)</li><li>• <code>after</code>: ISO 8601 date (e.g., "2020-01-01") - hint for results after this date (added to prompt context)</li><li>• <code>before</code>: ISO 8601 date (e.g., "2024-01-01") - hint for results before this date (added to prompt context)</li></ul>
<code>config</code>	An <code>asa_config</code> object for unified configuration, or <code>NULL</code> to use defaults
<code>agent</code>	An <code>asa_agent</code> object from <a href="#">initialize_agent</a> , or <code>NULL</code> to use the currently initialized agent
<code>verbose</code>	Print progress messages (default: <code>FALSE</code> )

## Details

This function provides the primary interface for running research tasks. For simple text responses, use `output_format = "text"`. For structured outputs, use `output_format = "json"` or specify field names to extract. For debugging and full trace access, use `output_format = "raw"`.

When temporal filtering is specified, the search tool's time filter is temporarily set for this task and restored afterward. Date hints (after/before) are appended to the prompt to guide the agent's search behavior.

**Value**

An `asa_result` object with:

- `prompt`: The original prompt
- `message`: The agent's response text
- `parsed`: Parsed output (list for JSON/field extraction, NULL for text/raw)
- `raw_output`: Full agent trace (always included, verbose for "raw" format)
- `elapsed_time`: Execution time in minutes
- `status`: "success" or "error"
- `trace`: Full execution trace (for "raw" `output_format`)
- `fold_count`: Number of memory folds (for "raw" `output_format`)

**See Also**

[initialize\\_agent](#), [run\\_task\\_batch](#), [asa\\_config](#), [temporal\\_options](#)

**Examples**

```
## Not run:
# Initialize agent first
agent <- initialize_agent(backend = "openai", model = "gpt-4.1-mini")

# Simple text query
result <- run_task(
  prompt = "What is the capital of France?",
  output_format = "text",
  agent = agent
)
print(result$message)

# JSON structured output
result <- run_task(
  prompt = "Find information about Albert Einstein and return JSON with
            fields: birth_year, death_year, nationality, field_of_study",
  output_format = "json",
  agent = agent
)
print(result$parsed)

# Raw output for debugging (includes full trace in asa_result)
result <- run_task(
  prompt = "Search for information",
  output_format = "raw",
  agent = agent
)
cat(result$trace) # View full agent trace

# With temporal filtering (past year only)
result <- run_task(
  prompt = "Find recent AI research breakthroughs",
  temporal = temporal_options(time_filter = "y"),
  agent = agent
)
```

```

# With date range hint
result <- run_task(
  prompt = "Find tech companies founded recently",
  temporal = list(
    time_filter = "y",
    after = "2020-01-01",
    before = "2024-01-01"
  ),
  agent = agent
)

# Using asa_config for unified configuration
config <- asa_config(
  backend = "openai",
  model = "gpt-4.1-mini",
  temporal = temporal_options(time_filter = "y")
)
result <- run_task(prompt, config = config)

## End(Not run)

```

**run\_task\_batch***Run Multiple Tasks in Batch***Description**

Executes multiple research tasks, optionally in parallel.

**Usage**

```
run_task_batch(
  prompts,
  output_format = "text",
  temporal = NULL,
  agent = NULL,
  parallel = FALSE,
  workers = 4L,
  progress = TRUE
)
```

**Arguments**

<b>prompts</b>	Character vector of task prompts, or a data frame with a 'prompt' column
<b>output_format</b>	Expected output format (applies to all tasks)
<b>temporal</b>	Named list for temporal filtering (applies to all tasks). See <a href="#">run_task</a> for details.
<b>agent</b>	An asa_agent object
<b>parallel</b>	Use parallel processing
<b>workers</b>	Number of parallel workers
<b>progress</b>	Show progress messages

**Value**

A list of `asa_result` objects, or if `prompts` was a data frame, the data frame with result columns added

**See Also**

[run\\_task](#), [configure\\_temporal](#)

**Examples**

```
## Not run:
prompts <- c(
  "What is the population of Tokyo?",
  "What is the population of New York?",
  "What is the population of London?"
)
results <- run_task_batch(prompts, agent = agent)

# With temporal filtering for all tasks
results <- run_task_batch(
  prompts,
  temporal = list(time_filter = "y"),
  agent = agent
)

## End(Not run)
```

**safe\_json\_parse**      *Safe JSON Parse*

**Description**

Attempts to parse JSON, returning NULL on failure.

**Usage**

`safe_json_parse(x)`

**Arguments**

<code>x</code>	JSON string
----------------	-------------

**Value**

Parsed R object or NULL

---

search_options	<i>Create Search Options</i>
----------------	------------------------------

---

### Description

Creates search configuration for controlling DuckDuckGo search behavior, including rate limiting, retry policies, and result limits. These options are used by the 4-tier search fallback system.

### Usage

```
search_options(  
    max_results = NULL,  
    timeout = NULL,  
    max_retries = NULL,  
    retry_delay = NULL,  
    backoff_multiplier = NULL,  
    inter_search_delay = NULL  
)
```

### Arguments

max_results	Maximum number of search results to return per query. Higher values provide more context but increase latency. Default: 10.
timeout	Timeout in seconds for individual search requests. Applies to each tier attempt separately. Default: 15.
max_retries	Maximum number of retry attempts when a search tier fails. After exhausting retries, the system falls back to the next tier. Default: 3.
retry_delay	Initial delay in seconds before the first retry. Subsequent retries use exponential backoff. Default: 2.
backoff_multiplier	Multiplier for exponential backoff between retries. E.g., with retry_delay=2 and multiplier=1.5, delays are 2s, 3s, 4.5s. Default: 1.5.
inter_search_delay	Minimum delay in seconds between consecutive searches. Helps avoid rate limiting from search providers. Default: 0.5.

### Details

The search system uses a 4-tier fallback architecture:

1. **PRIMP**: HTTP/2 with browser TLS fingerprint
2. **Selenium**: Headless browser for JS-rendered content
3. **DDGS**: Standard ddgs Python library
4. **Requests**: Raw POST to DuckDuckGo HTML endpoint

The retry/backoff settings apply within each tier. If all retries are exhausted, the system automatically falls back to the next tier.

### Value

An object of class `asa_search`

**See Also**

[asa\\_config](#), [configure\\_search](#)

**Examples**

```
## Not run:
# Default settings
search <- search_options()

# More aggressive settings for faster searches
search <- search_options(
  max_results = 5,
  timeout = 10,
  max_retries = 2
)

# Conservative settings for rate-limited environments
search <- search_options(
  inter_search_delay = 2.0,
  max_retries = 5,
  backoff_multiplier = 2.0
)

# Use with asa_config
config <- asa_config(
  backend = "openai",
  search = search_options(max_results = 15)
)

## End(Not run)
```

**summary.asa\_agent**      *Summary Method for asa\_agent Objects*

**Description**

Summary Method for asa\_agent Objects

**Usage**

```
## S3 method for class 'asa_agent'
summary(object, ...)
```

**Arguments**

object	An asa_agent object
...	Additional arguments (ignored)

**Value**

Invisibly returns a summary list

---

**summary.asa\_audit\_result**

*Summary Method for asa\_audit\_result Objects*

---

**Description**

Summary Method for asa\_audit\_result Objects

**Usage**

```
## S3 method for class 'asa_audit_result'  
summary(object, ...)
```

**Arguments**

object	An asa_audit_result object
...	Additional arguments (ignored)

**Value**

Invisibly returns a summary list

---

**summary.asa\_enumerate\_result**

*Summary Method for asa\_enumerate\_result Objects*

---

**Description**

Summary Method for asa\_enumerate\_result Objects

**Usage**

```
## S3 method for class 'asa_enumerate_result'  
summary(object, ...)
```

**Arguments**

object	An asa_enumerate_result object
...	Additional arguments (ignored)

**Value**

Invisibly returns a summary list

---

**summary.asa\_response**    *Summary Method for asa\_response Objects*

---

### Description

Summary Method for asa\_response Objects

### Usage

```
## S3 method for class 'asa_response'  
summary(object, show_trace = FALSE, ...)
```

### Arguments

object	An asa_response object
show_trace	Include full trace in output
...	Additional arguments (ignored)

### Value

Invisibly returns a summary list

---

**summary.asa\_result**    *Summary Method for asa\_result Objects*

---

### Description

Summary Method for asa\_result Objects

### Usage

```
## S3 method for class 'asa_result'  
summary(object, ...)
```

### Arguments

object	An asa_result object
...	Additional arguments (ignored)

### Value

Invisibly returns a summary list

---

**temporal\_options**      *Create Temporal Filtering Options*

---

**Description**

Creates a temporal filtering configuration for constraining search results by date. Supports DuckDuckGo time filters, date ranges, and strict verification modes.

**Usage**

```
temporal_options(  
    time_filter = NULL,  
    after = NULL,  
    before = NULL,  
    strictness = "best_effort",  
    use_wayback = FALSE  
)
```

**Arguments**

time_filter	DuckDuckGo time filter: "d" (day), "w" (week), "m" (month), "y" (year), or NULL for no filter
after	ISO 8601 date string (e.g., "2020-01-01") - results after this date
before	ISO 8601 date string (e.g., "2024-01-01") - results before this date
strictness	Verification level: "best_effort" (default) or "strict"
use_wayback	Use Wayback Machine for strict pre-date guarantees

**Details**

Temporal filtering can operate at different levels:

- **time\_filter**: DuckDuckGo native filter (fast, approximate)
- **after/before**: Date hints appended to prompts
- **strict**: Post-hoc verification of result dates
- **use\_wayback**: Uses Internet Archive for guaranteed historical data

**Value**

An object of class `asa_temporal`

**See Also**

[asa\\_config](#), [run\\_task](#)

## Examples

```
## Not run:
# Past year only
temporal <- temporal_options(time_filter = "y")

# Specific date range
temporal <- temporal_options(
  after = "2020-01-01",
  before = "2024-01-01"
)

# Strict historical verification
temporal <- temporal_options(
  before = "2015-01-01",
  strictness = "strict",
  use_wayback = TRUE
)

## End(Not run)
```

**truncate\_string**      *Truncate String*

## Description

Truncates a string to a maximum length, adding ellipsis if truncated.

## Usage

```
truncate_string(x, max_length = 100, ellipsis = "...")
```

## Arguments

x	Character string
max_length	Maximum length
ellipsis	String to append when truncated

## Value

Truncated string

---

```
write_csv.asa_enumerate_result
    Write asa_enumerate_result to CSV
```

---

## Description

Write asa\_enumerate\_result to CSV

## Usage

```
write_csv.asa_enumerate_result(x, file, include_provenance = FALSE, ...)
```

## Arguments

x	An asa_enumerate_result object
file	Path to output CSV file
include_provenance	Include provenance as additional columns
...	Additional arguments passed to write.csv

## Value

Invisibly returns the file path

# Index

- \* **internal**
  - .asa\_option, 5
  - .augment\_prompt\_temporal, 6
  - .build\_trace, 6
  - .close\_http\_clients, 7
  - .create\_agent, 7
  - .create\_http\_clients, 8
  - .create\_llm, 8
  - .create\_research\_config, 9
  - .create\_research\_graph, 9
  - .create\_tools, 9
  - .extract\_fields, 10
  - .extract\_json\_from\_trace, 10
  - .extract\_json\_object, 10
  - .extract\_response\_text, 11
  - .get\_default\_backend, 11
  - .get\_default\_conda\_env, 11
  - .get\_default\_model, 11
  - .get\_default\_workers, 12
  - .get\_extdata\_path, 12
  - .get\_local\_ip, 12
  - .get\_python\_path, 13
  - .handle\_response\_issues, 13
  - .import\_python\_module, 13
  - .import\_python\_packages, 14
  - .import\_research\_modules, 14
  - .invoke\_memory\_folding\_agent, 14
  - .invoke\_standard\_agent, 14
  - .is\_initialized, 15
  - .normalize\_schema, 15
  - .parse\_json\_response, 15
  - .process\_research\_results, 16
  - .resume\_research, 16
  - .run\_agent, 16
  - .run\_research, 17
  - .run\_research\_with\_progress, 17
  - .save\_checkpoint, 17
  - .stop\_validation, 18
  - .validate\_api\_key, 18
  - .validate\_asa\_agent, 18
  - .validate\_asa\_response, 19
  - .validate\_asa\_result, 19
  - .validate\_build\_backend, 19
  - .validate\_build\_prompt, 20
  - .validate\_choice, 20
  - .validate\_conda\_env, 20
  - .validate\_configure\_search, 21
  - .validate\_consistency, 21
  - .validate\_dataframe, 22
  - .validate\_initialize\_agent, 22
  - .validate\_logical, 23
  - .validate\_positive, 23
  - .validate\_process\_outputs, 23
  - .validate\_proxy\_url, 24
  - .validate\_range, 24
  - .validate\_required, 24
  - .validate\_research\_inputs, 25
  - .validate\_run\_agent, 25
  - .validate\_run\_task, 25
  - .validate\_run\_task\_batch, 26
  - .validate\_s3\_class, 26
  - .validate\_string, 27
  - .validate\_string\_vector, 27
  - .validate\_temporal, 28
  - .with\_temporal, 28
- asa-package, 5
- ASA\_API\_ENDPOINTS, 31
- ASA\_API\_KEY\_ENV\_VARS, 31
- ASA\_DEFAULT\_BACKEND, 36
- ASA\_DEFAULT\_BUDGET\_QUERIES, 36
- ASA\_DEFAULT\_BUDGET\_TIME, 36
- ASA\_DEFAULT\_BUDGET\_TOKENS, 37
- ASA\_DEFAULT\_CONDA\_ENV, 37
- ASA\_DEFAULT\_INTER\_SEARCH\_DELAY, 37
- ASA\_DEFAULT\_MAX\_RESULTS, 38
- ASA\_DEFAULT\_MAX\_RETRIES, 38
- ASA\_DEFAULT\_MAX\_ROUNDS, 38
- ASA\_DEFAULT\_MEMORY\_FOLDING, 39
- ASA\_DEFAULT\_MEMORY\_KEEP\_RECENT, 39
- ASA\_DEFAULT\_MEMORY\_THRESHOLD, 39
- ASA\_DEFAULT\_MODEL, 40
- ASA\_DEFAULT\_NOVELTY\_MIN, 40
- ASA\_DEFAULT\_NOVELTY\_WINDOW, 40
- ASA\_DEFAULT\_PAGE\_LOAD\_WAIT, 41
- ASA\_DEFAULT\_PLATEAU\_ROUNDS, 41
- ASA\_DEFAULT\_PROXY, 41

ASA\_DEFAULT\_RATE\_LIMIT, 42  
ASA\_DEFAULT\_TEMPERATURES, 42  
ASA\_DEFAULT\_TIMEOUT, 42  
ASA\_DEFAULT\_WIKI\_CHARS, 43  
ASA\_DEFAULT\_WIKI\_TOP\_K, 43  
ASA\_DEFAULT\_WORKERS, 43  
ASA\_OUTPUT\_FORMATS, 48  
ASA\_PRINT\_WIDTH, 48  
ASA\_RATE\_LIMIT\_WAIT, 49  
ASA\_RECURSION\_LIMIT\_FOLDING, 49  
ASA\_RECURSION\_LIMIT\_STANDARD, 49  
ASA\_STATUS\_ERROR, 51  
ASA\_STATUS\_SUCCESS, 51  
ASA\_SUPPORTED\_BACKENDS, 52  
ASA\_TEMPORAL\_STRICTNESS, 52  
ASA\_TIME\_FILTERS, 52  
ASA\_TRUNCATE\_LENGTH, 53  
clean\_whitespace, 55  
decode\_html, 59  
format\_duration, 62  
json\_escape, 66  
print2, 71  
safe\_json\_parse, 76  
truncate\_string, 82  
.asa\_option, 5  
.augment\_prompt\_temporal, 6  
.build\_trace, 6  
.close\_http\_clients, 7  
.create\_agent, 7  
.create\_http\_clients, 8  
.create\_llm, 8  
.create\_research\_config, 9  
.create\_research\_graph, 9  
.create\_tools, 9  
.extract\_fields, 10  
.extract\_json\_from\_trace, 10  
.extract\_json\_object, 10  
.extract\_response\_text, 11  
.get\_default\_backend, 11  
.get\_default\_conda\_env, 11  
.get\_default\_model, 11  
.get\_default\_workers, 12  
.get\_extdata\_path, 12  
.get\_local\_ip, 12  
.get\_python\_path, 13  
.handle\_response\_issues, 13  
.import\_python\_module, 13  
.import\_python\_packages, 14  
.import\_research\_modules, 14  
.invoke\_memory\_folding\_agent, 14  
.invoke\_standard\_agent, 14  
.is\_initialized, 15  
.normalize\_schema, 15  
.parse\_json\_response, 15  
.process\_research\_results, 16  
.resume\_research, 16  
.run\_agent, 16  
.run\_research, 17  
.run\_research\_with\_progress, 17  
.save\_checkpoint, 17  
.stop\_validation, 18  
.validate\_api\_key, 18  
.validate\_asa\_agent, 18  
.validate\_asa\_response, 19  
.validate\_asa\_result, 19  
.validate\_build\_backend, 19  
.validate\_build\_prompt, 20  
.validate\_choice, 20  
.validate\_conda\_env, 20  
.validate\_configure\_search, 21  
.validate\_consistency, 21  
.validate\_dataframe, 22  
.validate\_initialize\_agent, 22  
.validate\_logical, 23  
.validate\_positive, 23  
.validate\_process\_outputs, 23  
.validate\_proxy\_url, 24  
.validate\_range, 24  
.validate\_required, 24  
.validate\_research\_inputs, 25  
.validate\_run\_agent, 25  
.validate\_run\_task, 25  
.validate\_run\_task\_batch, 26  
.validate\_s3\_class, 26  
.validate\_string, 27  
.validate\_string\_vector, 27  
.validate\_temporal, 28  
.with\_temporal, 28  
  
as.data.frame.asa\_audit\_result, 29  
as.data.frame.asa\_enumerate\_result, 29  
as.data.frame.asa\_result, 30  
asa (asa-package), 5  
asa-package, 5  
asa\_agent, 30  
ASA\_API\_ENDPOINTS, 31  
ASA\_API\_KEY\_ENV\_VARS, 31  
asa\_audit, 31  
asa\_audit\_result, 33  
asa\_config, 34, 74, 78, 81  
ASA\_DEFAULT\_BACKEND, 36  
ASA\_DEFAULT\_BUDGET\_QUERIES, 36  
ASA\_DEFAULT\_BUDGET\_TIME, 36  
ASA\_DEFAULT\_BUDGET\_TOKENS, 37  
ASA\_DEFAULT\_CONDA\_ENV, 37

ASA\_DEFAULT\_INTER\_SEARCH\_DELAY, 37  
ASA\_DEFAULT\_MAX\_RESULTS, 38  
ASA\_DEFAULT\_MAX\_RETRIES, 38  
ASA\_DEFAULT\_MAX\_ROUNDS, 38  
ASA\_DEFAULT\_MEMORY\_FOLDING, 39  
ASA\_DEFAULT\_MEMORY\_KEEP\_RECENT, 39  
ASA\_DEFAULT\_MEMORY\_THRESHOLD, 39  
ASA\_DEFAULT\_MODEL, 40  
ASA\_DEFAULT\_NOVELTY\_MIN, 40  
ASA\_DEFAULT\_NOVELTY\_WINDOW, 40  
ASA\_DEFAULT\_PAGE\_LOAD\_WAIT, 41  
ASA\_DEFAULT\_PLATEAU\_ROUNDS, 41  
ASA\_DEFAULT\_PROXY, 41  
ASA\_DEFAULT\_RATE\_LIMIT, 42  
ASA\_DEFAULT\_TEMPERATURES, 42  
ASA\_DEFAULT\_TIMEOUT, 42  
ASA\_DEFAULT\_WIKI\_CHARS, 43  
ASA\_DEFAULT\_WIKI\_TOP\_K, 43  
ASA\_DEFAULT\_WORKERS, 43  
asa\_enumerate, 44, 59  
asa\_enumerate\_result, 47  
ASA\_OUTPUT\_FORMATS, 48  
ASA\_PRINT\_WIDTH, 48  
ASA\_RATE\_LIMIT\_WAIT, 49  
ASA\_RECURSION\_LIMIT\_FOLDING, 49  
ASA\_RECURSION\_LIMIT\_STANDARD, 49  
asa\_response, 50  
asa\_result, 50  
ASA\_STATUS\_ERROR, 51  
ASA\_STATUS\_SUCCESS, 51  
ASA\_SUPPORTED\_BACKENDS, 52  
ASA\_TEMPORAL\_STRICTNESS, 52  
ASA\_TIME\_FILTERS, 52  
ASA\_TRUNCATE\_LENGTH, 53  
  
build\_backend, 5, 53  
build\_prompt, 54  
  
check\_backend, 55  
clean\_whitespace, 55  
configure\_search, 56, 78  
configure\_search\_logging, 57  
configure\_temporal, 58, 76  
  
decode\_html, 59  
  
extract\_agent\_results, 59  
extract\_search\_snippets, 60  
extract\_search\_tiers, 61  
extract\_urls, 61  
extract\_wikipedia\_content, 62  
  
format\_duration, 62  
  
get\_agent, 63  
get\_tor\_ip, 63  
  
initialize\_agent, 5, 46, 64, 73, 74  
is\_tor\_running, 66  
  
json\_escape, 66  
  
print.asa\_agent, 67  
print.asa\_audit\_result, 67  
print.asa\_config, 68  
print.asa\_enumerate\_result, 68  
print.asa\_response, 69  
print.asa\_result, 69  
print.asa\_search, 70  
print.asa\_temporal, 70  
print2, 71  
process\_outputs, 71  
  
reset\_agent, 72  
rotate\_tor\_circuit, 72  
run\_task, 5, 16, 46, 59, 65, 73, 75, 76, 81  
run\_task\_batch, 5, 65, 74, 75  
  
safe\_json\_parse, 76  
search\_options, 35, 77  
summary.asa\_agent, 78  
summary.asa\_audit\_result, 79  
summary.asa\_enumerate\_result, 79  
summary.asa\_response, 80  
summary.asa\_result, 80  
  
temporal\_options, 35, 74, 81  
truncate\_string, 82  
  
write\_csv.asa\_enumerate\_result, 83