Package: gym (via r-universe)

October 27, 2024

Version 0.1.0

Title Provides Access to the OpenAI Gym API

Description OpenAI Gym is a open-source Python toolkit for developing and comparing reinforcement learning algorithms. This is a wrapper for the OpenAI Gym API, and enables access to an ever-growing variety of environments. For more details on OpenAI Gym, please see here: <https://github.com/openai/gym>. For more details on the OpenAI Gym API specification, please see here: <https://github.com/openai/gym-http-api>.

License MIT + file LICENSE

LazyData true

Depends R (>= 3.3.1)

Imports httr, jsonlite

URL https://github.com/paulhendricks/gym-R

BugReports https://github.com/paulhendricks/gym-R/issues

RoxygenNote 5.0.1

Suggests testthat

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

RemoteUrl https://github.com/paulhendricks/gym-r

RemoteRef HEAD

RemoteSha 52237a6de5d57dcf7dde2164435e910cba2287cb

Contents

`create_GymClient` *Create a GymClient instance.*

Description

This function instantiates a GymClient instance to integrate with an OpenAI Gym server.

Usage

```
create_GymClient(remote_base)
```

Arguments

`remote_base`	The URL of the OpenAI gym server. This value is usually "http://127.0.0.1:5000".

Value

An instance of class "GymClient"; this object has "remote_base" as an attribute.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)

## End(Not run)
```

env_action_space_contains

Evaluate whether an action is a member of an environments's action space.

Description

Evaluate whether an action is a member of an environments's action space.

Usage

```
env_action_space_contains(x, instance_id, action)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
instance_id	A short identifier (such as "3c657dbc") for the environment instance.
action	An action to take in the environment.

Value

A boolean atomic vector of length one indicating if the action is a member of an environments's action space.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
action <- env_action_space_sample(client, instance_id)
env_action_space_contains(client, instance_id, action)

## End(Not run)
```

env_action_space_info

Get information (name and dimensions/bounds) of the environments's action space.

Description

Get information (name and dimensions/bounds) of the environments's action space.

Usage

```
env_action_space_info(x, instance_id)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
instance_id	A short identifier (such as "3c657dbc") for the environment instance.

Value

A list containing "name" (such as "Discrete"), and additional dimensional info (such as "n") which varies from space to space.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
env_action_space_info(client, instance_id)

## End(Not run)
```

---

env_action_space_sample

Sample an action from the environments's action space.

Description

Sample an action from the environments's action space.

Usage

```
env_action_space_sample(x, instance_id)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
instance_id	A short identifier (such as "3c657dbc") for the environment instance.

Value

An action sampled from a space (such as "Discrete"), which varies from space to space.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
env_action_space_sample(client, instance_id)

## End(Not run)
```

`env_close` *Flush all monitor data to disk.*

Description

Flush all monitor data to disk.

Usage

```
env_close(x, instance_id)
```

Arguments

`x`	An instance of class "GymClient"; this object has "remote_base" as an attribute.
`instance_id`	A short identifier (such as "3c657dbc") for the environment instance.

Value

NULL.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
env_close(client, instance_id)

## End(Not run)
```

env_create *Create an instance of the specified environment.*

Description

Create an instance of the specified environment.

Usage

```
env_create(x, env_id)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
env_id	A short identifier (such as "3c657dbc") for the created environment instance. The instance_id is used in future API calls to identify the environment to be manipulated.

Value

A short identifier (such as "3c657dbc") for the created environment instance. The instance_id is used in future API calls to identify the environment to be manipulated.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
env_create(client, env_id)

## End(Not run)
```

env_list_all *List all environments running on the server.*

Description

List all environments running on the server.

Usage

```
env_list_all(x)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.

Value

A list mapping instance_id to env_id e.g. `list("3c657dbc" = "CartPole-v0")` for every env on the server.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_list_all(client)

## End(Not run)
```

`env_monitor_close` *Flush all monitor data to disk.*

Description

Flush all monitor data to disk.

Usage

```
env_monitor_close(x, instance_id)
```

Arguments

`x`	An instance of class "GymClient"; this object has "remote_base" as an attribute.
`instance_id`	A short identifier (such as "3c657dbc") for the environment instance.

Value

NULL.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
env_monitor_close(client, instance_id)

## End(Not run)
```

`env_monitor_start` *Start monitoring.*

Description

Start monitoring.

Usage

```
env_monitor_start(x, instance_id, directory, force = FALSE, resume = FALSE)
```

Arguments

`x`	An instance of class "GymClient"; this object has "remote_base" as an attribute.
`instance_id`	A short identifier (such as "3c657dbc") for the environment instance.
`directory`	The directory to write the training data to. Defaults to FALSE.
`force`	Clear out existing training data from this directory (by deleting every file prefixed with "openaigym"). Defaults to NULL.
`resume`	Retain the training data already in this directory, which will be merged with our new data. Defaults to FALSE.

Value

NULL.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
outdir <- "/tmp/random-agent-results"
env_monitor_start(client, instance_id, outdir, force = TRUE, resume = FALSE)

## End(Not run)
```

env_observation_space_info

Get information (name and dimensions/bounds) of the environment's observation space.

Description

Get information (name and dimensions/bounds) of the environment's observation space.

Usage

```
env_observation_space_info(x, instance_id)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
instance_id	A short identifier (such as "3c657dbc") for the environment instance.

Value

A list containing "name" (such as "Discrete"), and additional dimensional info (such as "n") which varies from space to space.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
env_observation_space_info(client, instance_id)

## End(Not run)
```

env_reset

Reset the state of the environment and return an initial observation.

Description

Reset the state of the environment and return an initial observation.

Usage

```
env_reset(x, instance_id)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
instance_id	A short identifier (such as "3c657dbc") for the environment instance.

Value

The initial observation of the space.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
env_reset(client, instance_id)

## End(Not run)
```

---

env_step *Step though an environment using an action.*

Description

Step though an environment using an action.

Usage

```
env_step(x, instance_id, action, render = FALSE)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
instance_id	A short identifier (such as "3c657dbc") for the environment instance.
action	An action to take in the environment.
render	Whether to render the environment. Defaults to FALSE.

Value

A list consisting of the following: action; an action to take in the environment, observation; an agent's observation of the current environment, reward; the amount of reward returned after previous action, done; whether the episode has ended, and info; a list containing auxiliary diagnostic information.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
env_id <- "CartPole-v0"
instance_id <- env_create(client, env_id)
action <- env_action_space_sample(client, instance_id)
env_step(client, instance_id, action)

## End(Not run)
```

---

get_request *Submit a GET request to an OpenAI Gym server.*

Description

Submit a GET request to an OpenAI Gym server.

Usage

```
get_request(x, route, data = NULL)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
route	The URL path or endpoint.
data	URL query arguments. Default value is NULL.

Value

If the response code is 200 or 204, a parsed response. Else, a server error or raised exception.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
route <- "/v1/envs/"
get_request(client, route)

## End(Not run)
```

---

gym *gym: Provides Access to the OpenAI Gym API*

Description

gym: Provides Access to the OpenAI Gym API

GymClient *Create a GymClient instance.*

Description

This function instantiates a GymClient instance to integrate with an OpenAI Gym server.

Usage

```
GymClient(remote_base)
```

Arguments

remote_base	The URL of the OpenAI gym server. This value is usually "http://127.0.0.1:5000".

Value

An instance of class "GymClient"; this object has "remote_base" as an attribute.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- GymClient(remote_base)

## End(Not run)
```

parse_server_error_or_raise_for_status *Parse the server error or raise for status.*

Description

Parse the server error or raise for status.

Usage

```
parse_server_error_or_raise_for_status(response)
```

Arguments

response	A response object from `httr::POST` or `httr::GET`.

Value

If the response code is 200 or 204, a parsed response. Else, a server error or raised exception.

Examples

```
## Not run:
b2 <- "http://httpbin.org/post"
response <- httr::POST(b2, body = "A simple text string")
parse_server_error_or_raise_for_status(response)

## End(Not run)
```

`post_request` *Submit a POST request to an OpenAI Gym server.*

Description

Submit a POST request to an OpenAI Gym server.

Usage

```
post_request(x, route, data = NULL)
```

Arguments

`x`	An instance of class "GymClient"; this object has "remote_base" as an attribute.
`route`	The URL path or endpoint.
`data`	URL query arguments. Default value is NULL.

Value

If the response code is 200 or 204, a parsed response. Else, a server error or raised exception.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
route <- "/v1/envs/"
env_id <- "CartPole-v0"
data <- list(env_id = env_id)
post_request(client, route, data)

## End(Not run)
```

print.GymClient *Represent a GymClient instance on the command line.*

Description

Represent a GymClient instance on the command line.

Usage

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

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
...	Further arguments passed to or from other methods.

Value

x A GymClient instance.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
print(client)

## End(Not run)
```

random_discrete_agent *A sample random discrete agent.*

Description

A sample random discrete agent.

Usage

```
random_discrete_agent(n)
```

Arguments

n	The number of discrete action spaces available.

Value

NULL.

Examples

```
agent <- random_discrete_agent(10)
```

`shutdown_server` *Request a server shutdown.*

Description

Request a server shutdown.

Usage

```
shutdown_server(x)
```

Arguments

`x`	An instance of class "GymClient"; this object has "remote_base" as an attribute.

Value

NULL Currently used by the integration tests to repeatedly create and destroy fresh copies of the server running in a separate thread.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
shutdown_server(client)

## End(Not run)
```

upload *Flush all monitor data to disk.*

Description

Flush all monitor data to disk.

Usage

```
upload(x, training_dir, api_key = NULL, algorithm_id = NULL)
```

Arguments

x	An instance of class "GymClient"; this object has "remote_base" as an attribute.
training_dir	A directory containing the results of a training run.
api_key	Your OpenAI API key.
algorithm_id	An arbitrary string indicating the paricular version of the algorithm (including choices of parameters) you are running.

Value

NULL.

Examples

```
## Not run:
remote_base <- "http://127.0.0.1:5000"
client <- create_GymClient(remote_base)
outdir <- "/tmp/random-agent-results"
upload(client, outdir)

## End(Not run)
```

Index