GNES Documentation

Release 0.0.34

Han Xiao, Feng Wang, Jianfeng Yan, Jie Fu

Aug 23, 2019

GNES Microservices API

GNES *jee-nes* is **Generic Neural Elastic Search**, a cloud-native semantic search system based on deep neural network.

GNES enables large-scale index and semantic search for **text-to-text**, **image-to-image**, **video-to-video** and *any-to-any* content form.

CHAPTER 1

Highlights

1.1 All Microservices in GNES

[32mGNES v0.0.34: Generic Neural Elastic Search[0m, a cloud-native semantic search system based on deep neural network. It enables large-scale index and semantic search for text-to-text, image-to-image, video-to-video and any content form. Visit [4m[36mhttps://gnes.ai[0m for tutorials and documentations.

```
usage: gnes [-h] [-v] [--verbose]
            {frontend,encode,index,route,preprocess,grpc,client,compose} ...
```

1.1.1 Named Arguments

-v, --version show program's version number and exit

--verbose turn on detailed logging for debug

Default: False

1.1.2 GNES sub-commands

use "gnes [sub-command] –help" to get detailed information about each sub-command

cli Possible choices: frontend, encode, index, route, preprocess, grpc, client, compose

1.1.3 Sub-commands:

frontend

start a frontend service

```
gnes frontend [-h] [--port_in PORT_IN] [--port_out PORT_OUT]
              [--host_in HOST_IN] [--host_out HOST_OUT]
              [--socket_in {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,
↪SUB_CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
              [--socket_out {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,
↪SUB_CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
              [--port_ctrl PORT_CTRL] [--timeout TIMEOUT]
              [--dump_interval DUMP_INTERVAL] [--read_only]
              [--parallel_backend {thread,process}]
              [--num_parallel NUM_PARALLEL]
              [--parallel_type {PUSH_BLOCK,PUSH_NONBLOCK,PUB_BLOCK,PUB_NONBLOCK}]
              [--grpc_host GRPC_HOST] [--grpc_port GRPC_PORT]
              [--max_message_size MAX_MESSAGE_SIZE]
              [--max_concurrency MAX_CONCURRENCY]
```

Named Arguments

--port_in port for input data, default a random port between [49152, 65536]

Default: 54827

--port_out port for output data, default a random port between [49152, 65536]

Default: 54671

--host_in host address for input

Default: “0.0.0.0”

--host_out host address for output

Default: “0.0.0.0”

--socket_in Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for input port

Default: PULL_BIND

--socket_out Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for output port

Default: PUSH_BIND

--port_ctrl port for controlling the service, default a random port between [49152, 65536]

Default: 63498

--timeout timeout (ms) of all communication, -1 for waiting forever

Default: -1

--dump_interval	serialize the service to a file every n seconds Default: 5
--read_only	do not allow the service to modify the model, dump_interval will be ignored Default: True
--parallel_backend	Possible choices: thread, process parallel backend of the service Default: "thread"
--num_parallel	number of parallel services running at the same time, *port_in* and *port_out* will be set to random, and routers will be added automatically when necessary Default: 1
--parallel_type	Possible choices: PUSH_BLOCK, PUSH_NONBLOCK, PUB_BLOCK, PUB_NONBLOCK parallel type of the concurrent services Default: PUSH_NONBLOCK
--grpc_host	host address of the grpc service Default: "0.0.0.0"
--grpc_port	host port of the grpc service Default: 8800
--max_message_size	maximum send and receive size for grpc server in (MB) Default: 100
--max_concurrency	maximum concurrent client allowed Default: 10

encode

start an encoder service

```
gnes encode [-h] [--port_in PORT_IN] [--port_out PORT_OUT] [--host_in HOST_IN]
            [--host_out HOST_OUT]
            [--socket_in {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
            [--socket_out {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
            [--port_ctrl PORT_CTRL] [--timeout TIMEOUT]
            [--dump_interval DUMP_INTERVAL] [--read_only]
            [--parallel_backend {thread,process}]
            [--num_parallel NUM_PARALLEL]
            [--parallel_type {PUSH_BLOCK,PUSH_NONBLOCK,PUB_BLOCK,PUB_NONBLOCK}]
            --yaml_path YAML_PATH [--py_path PY_PATH [PY_PATH ...]]
```

Named Arguments

--port_in	port for input data, default a random port between [49152, 65536]

	Default: 58782
--port_out	port for output data, default a random port between [49152, 65536]
	Default: 54374
--host_in	host address for input
	Default: "0.0.0.0"
--host_out	host address for output
	Default: "0.0.0.0"
--socket_in	Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT
	socket type for input port
	Default: PULL_BIND
--socket_out	Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT
	socket type for output port
	Default: PUSH_BIND
--port_ctrl	port for controlling the service, default a random port between [49152, 65536]
	Default: 54044
--timeout	timeout (ms) of all communication, -1 for waiting forever
	Default: -1
--dump_interval	serialize the service to a file every n seconds
	Default: 5
--read_only	do not allow the service to modify the model, dump_interval will be ignored
	Default: False
--parallel_backend	Possible choices: thread, process
	parallel backend of the service
	Default: "thread"
--num_parallel	number of parallel services running at the same time, *port_in* and *port_out* will be set to random, and routers will be added automatically when necessary
	Default: 1
--parallel_type	Possible choices: PUSH_BLOCK, PUSH_NONBLOCK, PUB_BLOCK, PUB_NONBLOCK
	parallel type of the concurrent services
	Default: PUSH_NONBLOCK
--yaml_path	yaml config of the service, it should be a readable stream, or a valid file path, or a supported class name.
--py_path	the file path(s) of an external python module(s).

index

start an indexer service

```
gnes index [-h] [--port_in PORT_IN] [--port_out PORT_OUT] [--host_in HOST_IN]
           [--host_out HOST_OUT]
           [--socket_in {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
           [--socket_out {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
           [--port_ctrl PORT_CTRL] [--timeout TIMEOUT]
           [--dump_interval DUMP_INTERVAL] [--read_only]
           [--parallel_backend {thread,process}] [--num_parallel NUM_PARALLEL]
           [--parallel_type {PUSH_BLOCK,PUSH_NONBLOCK,PUB_BLOCK,PUB_NONBLOCK}]
           --yaml_path YAML_PATH [--py_path PY_PATH [PY_PATH ...]]
```

Named Arguments

--port_in port for input data, default a random port between [49152, 65536]

Default: 55592

--port_out port for output data, default a random port between [49152, 65536]

Default: 55594

--host_in host address for input

Default: “0.0.0.0”

--host_out host address for output

Default: “0.0.0.0”

--socket_in Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for input port

Default: PULL_CONNECT

--socket_out Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for output port

Default: PUB_BIND

--port_ctrl port for controlling the service, default a random port between [49152, 65536]

Default: 61530

--timeout timeout (ms) of all communication, -1 for waiting forever

Default: -1

--dump_interval serialize the service to a file every n seconds

Default: 5

--read_only	do not allow the service to modify the model, dump_interval will be ignored Default: False
--parallel_backend	Possible choices: thread, process parallel backend of the service Default: "thread"
--num_parallel	number of parallel services running at the same time, *port_in* and *port_out* will be set to random, and routers will be added automatically when necessary Default: 1
--parallel_type	Possible choices: PUSH_BLOCK, PUSH_NONBLOCK, PUB_BLOCK, PUB_NONBLOCK parallel type of the concurrent services Default: PUSH_NONBLOCK
--yaml_path	yaml config of the service, it should be a readable stream, or a valid file path, or a supported class name.
--py_path	the file path(s) of an external python module(s).

route

start a router service

```
gnes route [-h] [--port_in PORT_IN] [--port_out PORT_OUT] [--host_in HOST_IN]
           [--host_out HOST_OUT]
           [--socket_in {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
           [--socket_out {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
           [--port_ctrl PORT_CTRL] [--timeout TIMEOUT]
           [--dump_interval DUMP_INTERVAL] [--read_only]
           [--parallel_backend {thread,process}] [--num_parallel NUM_PARALLEL]
           [--parallel_type {PUSH_BLOCK,PUSH_NONBLOCK,PUB_BLOCK,PUB_NONBLOCK}]
           --yaml_path YAML_PATH [--py_path PY_PATH [PY_PATH ...]]
           [--num_part NUM_PART]
```

Named Arguments

--port_in	port for input data, default a random port between [49152, 65536] Default: 60303
--port_out	port for output data, default a random port between [49152, 65536] Default: 58761
--host_in	host address for input Default: "0.0.0.0"
--host_out	host address for output Default: "0.0.0.0"

--socket_in	Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT socket type for input port Default: PULL_BIND
--socket_out	Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT socket type for output port Default: PUSH_BIND
--port_ctrl	port for controlling the service, default a random port between [49152, 65536] Default: 54290
--timeout	timeout (ms) of all communication, -1 for waiting forever Default: -1
--dump_interval	serialize the service to a file every n seconds Default: 5
--read_only	do not allow the service to modify the model, dump_interval will be ignored Default: True
--parallel_backend	Possible choices: thread, process parallel backend of the service Default: “thread”
--num_parallel	number of parallel services running at the same time, *port_in* and *port_out* will be set to random, and routers will be added automatically when necessary Default: 1
--parallel_type	Possible choices: PUSH_BLOCK, PUSH_NONBLOCK, PUB_BLOCK, PUB_NONBLOCK parallel type of the concurrent services Default: PUSH_NONBLOCK
--yaml_path	yaml config of the service, it should be a readable stream, or a valid file path, or a supported class name.
--py_path	the file path(s) of an external python module(s).
--num_part	explicitly set the number of parts of message

preprocess

start a preprocessor service

```
gnes preprocess [-h] [--port_in PORT_IN] [--port_out PORT_OUT]
                [--host_in HOST_IN] [--host_out HOST_OUT]
                [--socket_in {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,
↪SUB_CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
```

(continues on next page)

(continued from previous page)

```
                [--socket_out {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,
↪SUB_CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
                [--port_ctrl PORT_CTRL] [--timeout TIMEOUT]
                [--dump_interval DUMP_INTERVAL] [--read_only]
                [--parallel_backend {thread,process}]
                [--num_parallel NUM_PARALLEL]
                [--parallel_type {PUSH_BLOCK,PUSH_NONBLOCK,PUB_BLOCK,PUB_NONBLOCK}]
                --yaml_path YAML_PATH [--py_path PY_PATH [PY_PATH ...]]
```

Named Arguments

--port_in port for input data, default a random port between [49152, 65536]

Default: 54281

--port_out port for output data, default a random port between [49152, 65536]

Default: 50018

--host_in host address for input

Default: "0.0.0.0"

--host_out host address for output

Default: "0.0.0.0"

--socket_in Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for input port

Default: PULL_BIND

--socket_out Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for output port

Default: PUSH_BIND

--port_ctrl port for controlling the service, default a random port between [49152, 65536]

Default: 64492

--timeout timeout (ms) of all communication, -1 for waiting forever

Default: -1

--dump_interval serialize the service to a file every n seconds

Default: 5

--read_only do not allow the service to modify the model, dump_interval will be ignored

Default: True

--parallel_backend Possible choices: thread, process

parallel backend of the service

Default: "thread"

--num_parallel number of parallel services running at the same time, *port_in* and *port_out* will be set to random, and routers will be added automatically when necessary

Default: 1

--parallel_type Possible choices: PUSH_BLOCK, PUSH_NONBLOCK, PUB_BLOCK, PUB_NONBLOCK

parallel type of the concurrent services

Default: PUSH_NONBLOCK

--yaml_path yaml config of the service, it should be a readable stream, or a valid file path, or a supported class name.

--py_path the file path(s) of an external python module(s).

grpc

start a general purpose grpc service

```
gnes grpc [-h] [--port_in PORT_IN] [--port_out PORT_OUT] [--host_in HOST_IN]
          [--host_out HOST_OUT]
          [--socket_in {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
          [--socket_out {PULL_BIND,PULL_CONNECT,PUSH_BIND,PUSH_CONNECT,SUB_BIND,SUB_
↪CONNECT,PUB_BIND,PUB_CONNECT,PAIR_BIND,PAIR_CONNECT}]
          [--port_ctrl PORT_CTRL] [--timeout TIMEOUT]
          [--dump_interval DUMP_INTERVAL] [--read_only]
          [--parallel_backend {thread,process}] [--num_parallel NUM_PARALLEL]
          [--parallel_type {PUSH_BLOCK,PUSH_NONBLOCK,PUB_BLOCK,PUB_NONBLOCK}]
          [--grpc_host GRPC_HOST] [--grpc_port GRPC_PORT]
          [--max_message_size MAX_MESSAGE_SIZE] --pb2_path PB2_PATH
          --pb2_grpc_path PB2_GRPC_PATH --stub_name STUB_NAME --api_name
          API_NAME
```

Named Arguments

--port_in port for input data, default a random port between [49152, 65536]

Default: 60969

--port_out port for output data, default a random port between [49152, 65536]

Default: 50871

--host_in host address for input

Default: “0.0.0.0”

--host_out host address for output

Default: “0.0.0.0”

--socket_in Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT

socket type for input port

Default: PULL_BIND

--socket_out	Possible choices: PULL_BIND, PULL_CONNECT, PUSH_BIND, PUSH_CONNECT, SUB_BIND, SUB_CONNECT, PUB_BIND, PUB_CONNECT, PAIR_BIND, PAIR_CONNECT socket type for output port Default: PUSH_BIND
--port_ctrl	port for controlling the service, default a random port between [49152, 65536] Default: 52356
--timeout	timeout (ms) of all communication, -1 for waiting forever Default: -1
--dump_interval	serialize the service to a file every n seconds Default: 5
--read_only	do not allow the service to modify the model, dump_interval will be ignored Default: False
--parallel_backend	Possible choices: thread, process parallel backend of the service Default: "thread"
--num_parallel	number of parallel services running at the same time, *port_in* and *port_out* will be set to random, and routers will be added automatically when necessary Default: 1
--parallel_type	Possible choices: PUSH_BLOCK, PUSH_NONBLOCK, PUB_BLOCK, PUB_NONBLOCK parallel type of the concurrent services Default: PUSH_NONBLOCK
--grpc_host	host address of the grpc service Default: "0.0.0.0"
--grpc_port	host port of the grpc service Default: 8800
--max_message_size	maximum send and receive size for grpc server in (MB) Default: 100
--pb2_path	the path of the python file protocol buffer compiler
--pb2_grpc_path	the path of the python file generated by the gRPC Python protocol compiler plugin
--stub_name	the name of the gRPC Stub
--api_name	the api name for calling the stub

client

start a GNES client of the selected type

```
gnes client [-h] {http,cli,benchmark} ...
```

GNES client sub-commands

use "gnes client [sub-command] –help" to get detailed information about each client sub-command

client Possible choices: http, cli, benchmark

Sub-commands:

http

start a client that allows HTTP requests as input

```
gnes client http [-h] [--grpc_host GRPC_HOST] [--grpc_port GRPC_PORT]
                 [--max_message_size MAX_MESSAGE_SIZE] [--http_port HTTP_PORT]
                 [--http_host HTTP_HOST] [--max_workers MAX_WORKERS]
                 [--top_k TOP_K] [--batch_size BATCH_SIZE]
```

Named Arguments

--grpc_host host address of the grpc service

Default: "0.0.0.0"

--grpc_port host port of the grpc service

Default: 8800

--max_message_size maximum send and receive size for grpc server in (MB)

Default: 100

--http_port http port to deploy the service

Default: 80

--http_host http host to deploy the service

Default: "0.0.0.0"

--max_workers max workers to deal with the message

Default: 100

--top_k default top_k for query mode

Default: 10

--batch_size batch size for feed data for train mode

Default: 2560

cli

start a client that allows stdin as input

```
gnes client cli [-h] [--grpc_host GRPC_HOST] [--grpc_port GRPC_PORT]
                [--max_message_size MAX_MESSAGE_SIZE]
                [--txt_file TXT_FILE | --image_zip_file IMAGE_ZIP_FILE | --video_zip_
↪file VIDEO_ZIP_FILE]
                [--batch_size BATCH_SIZE] --mode {index,query,train}
                [--top_k TOP_K] [--start_doc_id START_DOC_ID]
```

Named Arguments

--grpc_host	host address of the grpc service
	Default: "0.0.0.0"
--grpc_port	host port of the grpc service
	Default: 8800
--max_message_size	maximum send and receive size for grpc server in (MB)
	Default: 100
--txt_file	text file to be used, each line is a doc/query
	Default: <_io.TextIOWrapper name='<stdin>' mode='r' encoding='UTF-8'>
--image_zip_file	image zip file to be used, consists of multiple images
--video_zip_file	video zip file to be used, consists of multiple videos
--batch_size	the size of the request to split
	Default: 100
--mode	Possible choices: index, query, train
	the mode of the client and the server
--top_k	top_k results returned in the query mode
	Default: 10
--start_doc_id	the start number of doc id
	Default: 0

benchmark

start a client for benchmark and unittest

```
gnes client benchmark [-h] [--grpc_host GRPC_HOST] [--grpc_port GRPC_PORT]
                      [--max_message_size MAX_MESSAGE_SIZE]
                      [--batch_size BATCH_SIZE]
                      [--request_length REQUEST_LENGTH]
                      [--num_requests NUM_REQUESTS]
```

Named Arguments

--grpc_host	host address of the grpc service
	Default: "0.0.0.0"
--grpc_port	host port of the grpc service
	Default: 8800
--max_message_size	maximum send and receive size for grpc server in (MB)
	Default: 100
--batch_size	the size of the request to split
	Default: 64
--request_length	binary string length of each request
	Default: 1024
--num_requests	number of total requests
	Default: 128

compose

start a GNES Board to visualize YAML configs

```
gnes compose [-h] [--port PORT] [--name NAME] [--yaml_path YAML_PATH]
             [--html_path HTML_PATH] [--shell_path SHELL_PATH]
             [--swarm_path SWARM_PATH] [--k8s_path K8S_PATH]
             [--graph_path GRAPH_PATH]
             [--shell_log_redirect SHELL_LOG_REDIRECT] [--mermaid_leftright]
             [--docker_img DOCKER_IMG] [--flask | --serve]
             [--http_port HTTP_PORT]
```

Named Arguments

--port	host port of the grpc service
	Default: 8800
--name	name of the instance
	Default: "GNES app"
--yaml_path	yaml config of the service
	Default: <_io.BufferedReader name='/home/docs/checkouts/readthedocs.org/user_builds/gnes/checkouts/ example.yml'>
--html_path	output path of the HTML file, will contain all possible generations
--shell_path	output path of the shell-based starting script
--swarm_path	output path of the docker-compose file for Docker Swarm
--k8s_path	output path of the docker-compose file for Docker Swarm
--graph_path	output path of the mermaid graph file

--shell_log_redirect	the file path for redirecting shell output. when not given, the output will be flushed to stdout
--mermaid_leftright	showing the flow in left-to-right manner rather than top down Default: False
--docker_img	the docker image used in Docker Swarm & Kubernetes Default: "gnes/gnes:latest-alpine"
--flask	start a Flask server and serve the composer in interactive mode, aka GNES board Default: False
--serve	start a basic HTTP server and serve the composer in interactive mode, aka GNES board Default: False
--http_port	server port for receiving HTTP requests Default: 8080

1.2 gnes package

1.2.1 Subpackages

gnes.base package

Module contents

class gnes.base.**TrainableBase**(**args*, ***kwargs*)
Bases: object

The base class for preprocessor, encoder, indexer and router

close()
Release the resources as model is destroyed

dump(*filename: str = None*) → None
Serialize the object to a binary file :param filename: file path of the serialized file, if not given then *self.dump_full_path* is used

dump_full_path
Get the binary dump path

Returns

dump_yaml(*filename: str = None*) → None
Serialize the object to a yaml file :param filename: file path of the yaml file, if not given then *self.dump_yaml_path* is used

classmethod from_yaml(*constructor*, *node*, *stop_on_import_error=False*)

static load(*filename: str = None*) → T

classmethod load_yaml(*filename: Union[str, TextIO]*) → T

post_init()
Declare class attributes/members that can not be serialized in standard way

classmethod pre_init()

store_args_kwargs = False

classmethod to_yaml(*representer*, *data*)

train(**args*, ***kwargs*)
Train the model, need to be overrided

yaml_full_path
Get the file path of the yaml config :return:

class gnes.base.**CompositionalTrainableBase**(**args*, ***kwargs*)
Bases: *gnes.base.TrainableBase*

close()
Release the resources as model is destroyed

components

classmethod from_yaml(*constructor*, *node*)

is_pipeline

classmethod to_yaml(*representer*, *data*)

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.cli package

Submodules

gnes.cli.api module

gnes.cli.api.**client**(*args*)

gnes.cli.api.**compose**(*args*)

gnes.cli.api.**encode**(*args*)

gnes.cli.api.**frontend**(*args*)

gnes.cli.api.**grpc**(*args*)

gnes.cli.api.**index**(*args*)

gnes.cli.api.**preprocess**(*args*)

gnes.cli.api.**route**(*args*)

gnes.cli.parser module

gnes.cli.parser.**get_main_parser**()

gnes.cli.parser.**resolve_py_path**(*path*)

gnes.cli.parser.**resolve_yaml_path**(*path*)

gnes.cli.parser.**set_base_parser**()

gnes.cli.parser.**set_client_benchmark_parser**(*parser=None*)

gnes.cli.parser.**set_client_cli_parser**(*parser=None*)

gnes.cli.parser.**set_client_http_parser**(*parser=None*)

gnes.cli.parser.**set_composer_flask_parser**(*parser=None*)

gnes.cli.parser.**set_composer_parser**(*parser=None*)

gnes.cli.parser.**set_encoder_parser**(*parser=None*)

gnes.cli.parser.**set_frontend_parser**(*parser=None*)

gnes.cli.parser.**set_grpc_service_parser**(*parser=None*)

gnes.cli.parser.**set_indexer_parser**(*parser=None*)

gnes.cli.parser.**set_preprocessor_parser**(*parser=None*)

gnes.cli.parser.**set_router_parser**(*parser=None*)

gnes.cli.parser.**set_service_parser**(*parser=None*)

Module contents

gnes.cli.**main**()

gnes.client package

Submodules

gnes.client.base module

class gnes.client.base.**ZmqClient**(*args*)

Bases: object

close()

recv_message(*timeout: int = -1*) → gnes_pb2.Message

send_message(*message: gnes_pb2.Message*, *timeout: int = -1*)

gnes.client.benchmark module

class gnes.client.benchmark.**BenchmarkClient**(*args*)

Bases: object

gnes.client.cli module

class gnes.client.cli.**CLIClient**(*args*)

Bases: object

get_channel()

index(*all_bytes: List[bytes], stub*)

query(*all_bytes: List[bytes], stub*)

read_all() → List[bytes]

train(*all_bytes: List[bytes], stub*)

use_channel()

class gnes.client.cli.**ProgressBar**(*all_bytes: List[bytes], batch_size: int, bar_len: int = 20, task_name: str = ''*)

Bases: object

update()

gnes.client.http module

class gnes.client.http.**HttpClient**(*args=None*)

Bases: object

start()

Module contents

gnes.composer package

Submodules

gnes.composer.base module

class gnes.composer.base.**YamlComposer**(*args*)

Bases: object

class Layer(*layer_id: int = 0*)

Bases: object

append(*comp*)

default_values = {'image': None, 'income': 'pull', 'name': None, 'py_path': Nor

get_component_name

static get_value(*comp: Dict, key: str*)

is_heto_single_component

is_homo_multi_component

is_homogenous

is_single_component

add_comp(*comp: Dict*) → None

add_layer(*layer: Layer = None*) → None

build_all()

static build_dockerswarm(*all_layers: List[YamlComposer.Layer], docker_img: str = 'gnes/gnes:latest-alpine', volumes: Dict = None, networks: Dict = None*) → str

static build_html(*generate_dict: Dict[str, str]*) → str

static build_kubernetes(*all_layers: List[YamlComposer.Layer], *args, **kwargs*)

build_layers() → List[gnes.composer.base.YamlComposer.Layer]

static build_mermaid(*all_layers: List[YamlComposer.Layer], mermaid_leftright: bool = False*) → str

static build_shell(*all_layers: List[YamlComposer.Layer], log_redirect: str = None*) → str

check_fields(*comp: Dict*) → bool

comp2args = {'Encoder': Namespace(dump_interval=5, host_in='0.0.0.0', host_out='0.0.0

comp2file = {'Encoder': 'encode', 'Frontend': 'frontend', 'Indexer': 'index', 'Prep

gnes.composer.base.**parse_http_data**(*data*, *args*)

gnes.composer.flask module

class gnes.composer.flask.**YamlComposerFlask**(*args*)

Bases: object

run()

gnes.composer.http module

class gnes.composer.http.**YamlComposerHttp**(*args*)

Bases: object

run()

Module contents

gnes.encoder package

Subpackages

gnes.encoder.audio package

Submodules

gnes.encoder.audio.mfcc module

class gnes.encoder.audio.mfcc.**MfccEncoder**(*n_mfcc: int = 13*, *sample_rate: int = 16000*, *max_length: int = 100*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseAudioEncoder*

batch_size = 64

encode(*data: List[np.array], *args, **kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)

Train the model, need to be overrided

Module contents

gnes.encoder.image package

Subpackages

gnes.encoder.image.cvae_cores package

Submodules

gnes.encoder.image.cvae_cores.model module

Module contents

gnes.encoder.image.inception_cores package

Submodules

gnes.encoder.image.inception_cores.inception_utils module

gnes.encoder.image.inception_cores.inception_v4 module

Module contents

Submodules

gnes.encoder.image.cvae module

class gnes.encoder.image.cvae.**CVAEEncoder**(*model_dir: str*, *latent_dim: int = 300*, *select_method: str = 'MEAN'*, *l2_normalize: bool = False*, *use_gpu: bool = True*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseImageEncoder*

batch_size = 64

encode(*img: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.image.inception module

class gnes.encoder.image.inception.**TFInceptionEncoder**(*model_dir: str*, *select_layer: str = 'PreLogitsFlatten'*, *use_cuda: bool = False*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseImageEncoder*

batch_size = 64

encode(*img: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.image.onnx module

class gnes.encoder.image.onnx.**BaseONNXImageEncoder**(*model_name: str*, *model_dir: str*, *use_cuda: bool = False*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseImageEncoder*

batch_size = 64

encode(*img: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.image.torchvision module

class gnes.encoder.image.torchvision.**TorchvisionEncoder**(*model_name: str*, *layers: List[str]*, *model_dir: str*, *use_cuda: bool = False*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseImageEncoder*

batch_size = 64

encode(*img: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.encoder.numeric package

Submodules

gnes.encoder.numeric.hash module

class gnes.encoder.numeric.hash.**HashEncoder**(*num_bytes: int*, *num_bits: int = 8*, *num_idx: int = 3*, *kmeans_clusters: int = 100*, *method: str = 'product_uniform'*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseNumericEncoder*

batch_size = 2048

encode(*vecs: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

hash(*vecs*)

pred_kmeans(*vecs*)

ran_gen()

train(*vecs: numpy.ndarray*, **args*, ***kwargs*)
Train the model, need to be overrided

train_kmeans(*vecs*)

gnes.encoder.numeric.pca module

class gnes.encoder.numeric.pca.**PCALocalEncoder**(*output_dim: int*, *num_locals: int*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseNumericEncoder*

batch_size = 2048

encode(*vecs: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

train(*vecs: numpy.ndarray*, **args*, ***kwargs*) → None
Train the model, need to be overrided

gnes.encoder.numeric.pq module

class gnes.encoder.numeric.pq.**PQEncoder**(*num_bytes: int*, *cluster_per_byte: int = 255*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseBinaryEncoder*

batch_size = 2048

encode(*vecs: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

train(*vecs: numpy.ndarray*, **args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.numeric.tf_pq module

class gnes.encoder.numeric.tf_pq.**TFPQEncoder**(*num_bytes: int*, *cluster_per_byte: int = 255*, **args*, ***kwargs*)

Bases: *gnes.encoder.numeric.pq.PQEncoder*

batch_size = 8192

close()
Release the resources as model is destroyed

encode(*vecs: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

classmethod pre_init()

train(*vecs: numpy.ndarray*, **args*, ***kwargs*)

gnes.encoder.numeric.vlad module

class gnes.encoder.numeric.vlad.**VladEncoder**(*num_clusters: int*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseNumericEncoder*

batch_size = 2048

encode(*vecs: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

kmeans_pred(*vecs*)

kmeans_train(*vecs*)

train(*vecs: numpy.ndarray*, **args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.encoder.text package

Submodules

gnes.encoder.text.bert module

class gnes.encoder.text.bert.**BertEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

close()
Release the resources as model is destroyed

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

store_args_kwargs = True

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.text.bert.**BertEncoderServer**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

close()
Release the resources as model is destroyed

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

store_args_kwargs = True

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.text.bert.**BertEncoderWithServer**(**args*, ***kwargs*)
Bases: *gnes.base.CompositionalTrainableBase*

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.text.elmo module

class gnes.encoder.text.elmo.**ElmoEncoder**(*model_dir: str*, *pooling_layer: int = -1*, *pooling_strategy: str = 'REDUCE_MEAN'*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

batch_size = 64

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.text.flair module

class gnes.encoder.text.flair.**FlairEncoder**(*model_name: str = 'multi-forward-fast'*, *pooling_strategy: str = 'REDUCE_MEAN'*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.text.gpt module

class gnes.encoder.text.gpt.**GPT2Encoder**(*model_dir: str*, *use_cuda: bool = False*, *pooling_strategy: str = 'REDUCE_MEAN'*, **args*, ***kwargs*)
Bases: *gnes.encoder.text.gpt.GPTEncoder*

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.text.gpt.**GPTEncoder**(*model_dir: str*, *use_cuda: bool = False*, *pooling_strategy: str = 'REDUCE_MEAN'*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

batch_size = 64

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.text.torch_transformers module

class gnes.encoder.text.torch_transformers.**TorchTransformersEncoder**(*model_dir: str*, *model_name: str*, *tokenizer_name: str*, *use_cuda: bool = False*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.encoder.text.w2v module

class gnes.encoder.text.w2v.**Word2VecEncoder**(*model_dir: str*, *skiprows: int = 1*, *dimension: int = 300*, *pooling_strategy: str = 'REDUCE_MEAN'*, **args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseTextEncoder*

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

is_trained = True

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.encoder.video package

Subpackages

gnes.encoder.video.mixture_core package

Submodules

gnes.encoder.video.mixture_core.model module

Module contents

Submodules

gnes.encoder.video.incep_mixture module

class gnes.encoder.video.incep_mixture.**IncepMixtureEncoder**(*model_dir_inception: str*, *model_dir_mixture: str*, *select_layer: str = 'PreLogitsFlatten'*, *use_cuda: bool = False*, *feature_size: int = 300*, *vocab_size: int = 28*, *cluster_size: int = 256*, *method: str = 'fvnet'*, *input_size: int = 1536*, *vocab_size_2: int = 174*, *max_frames: int = 30*, *multitask_method: str = 'Attention'*, **args*, ***kwargs*)

Bases: *gnes.encoder.base.BaseVideoEncoder*

batch_size = 64

encode(*data: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

Submodules

gnes.encoder.base module

class gnes.encoder.base.**BaseAudioEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseEncoder*

encode(*data: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**BaseBinaryEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseEncoder*

encode(*data: numpy.ndarray*, **args*, ***kwargs*) → bytes

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**BaseEncoder**(**args*, ***kwargs*)
Bases: *gnes.base.TrainableBase*

encode(*data: Any*, **args*, ***kwargs*) → Any

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**BaseImageEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseEncoder*

encode(*img: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**BaseNumericEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseEncoder*

encode(*text: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**BaseTextEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseEncoder*

encode(*text: List[str], *args, **kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**BaseVideoEncoder**(**args*, ***kwargs*)
Bases: *gnes.encoder.base.BaseEncoder*

encode(*data: List[np.ndarray], *args, **kwargs*) → numpy.ndarray

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.encoder.base.**PipelineEncoder**(**args*, ***kwargs*)
Bases: *gnes.base.CompositionalTrainableBase*

encode(*data: Any*, **args*, ***kwargs*) → Any

train(*data*, **args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.indexer package

Subpackages

gnes.indexer.fulltext package

Submodules

gnes.indexer.fulltext.filesys module

class gnes.indexer.fulltext.filesys.**DirectoryIndexer**(*data_path: str*, *keep_na_doc: bool = True*, *file_suffix: str = 'gif'*, **args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseTextIndexer*

add(*keys: List[int], docs: List[gnes_pb2.Document], *args, **kwargs*)
write GIFs of each document into disk folder structure: /data_path/doc_id/0.gif, 1.gif... :param keys: list of doc id :param docs: list of docs

query(*keys: List[int], *args, **kwargs*) → List[gnes_pb2.Document]

Parameters **keys** – list of doc id

Returns list of documents whose chunks field contain all the GIFs of this doc(one GIF per chunk)

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.indexer.fulltext.leveldb module

class gnes.indexer.fulltext.leveldb.**AsyncLVDBIndexer**(*data_path: str*, *keep_na_doc: bool = True*, *drop_raw_bytes: bool = False*, *drop_chunk_blob: bool = False*, **args*, ***kwargs*)
Bases: *gnes.indexer.fulltext.leveldb.LVDBIndexer*

add(*keys: List[int], docs: List[gnes_pb2.Document], *args, **kwargs*)

close()
Release the resources as model is destroyed

post_init()
Declare class attributes/members that can not be serialized in standard way

query(**args*, ***kwargs*) → List[Any]

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.fulltext.leveldb.**LVDBIndexer**(*data_path: str*, *keep_na_doc: bool = True*, *drop_raw_bytes: bool = False*, *drop_chunk_blob: bool = False*, **args*, ***kwargs*)

Bases: *gnes.indexer.base.BaseTextIndexer*

add(*keys: List[int], docs: List[gnes_pb2.Document], *args, **kwargs*)

close()
Release the resources as model is destroyed

post_init()
Declare class attributes/members that can not be serialized in standard way

query(*keys: List[int], *args, **kwargs*) → List[gnes_pb2.Document]

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.indexer.vector package

Subpackages

gnes.indexer.vector.bindexer package

Submodules

gnes.indexer.vector.bindexer.bindexer module

Module contents

gnes.indexer.vector.hbindexer package

Submodules

gnes.indexer.vector.hbindexer.hbindexer module

Module contents

Submodules

gnes.indexer.vector.annoy module

class gnes.indexer.vector.annoy.**AnnoyIndexer**(*num_dim: int*, *data_path: str*, *metric: str = 'angular'*, *n_trees=10*, **args*, ***kwargs*)

Bases: *gnes.indexer.base.BaseVectorIndexer*

add(*keys: List[Tuple[int, Any]], vectors: numpy.ndarray, weights: List[float], *args, **kwargs*)

normalize_score(*score: List[float], metrics: str, *args, **kwargs*) → List[float]

post_init()
Declare class attributes/members that can not be serialized in standard way

query()

size

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.indexer.vector.faiss module

class gnes.indexer.vector.faiss.**FaissIndexer**(*num_dim: int*, *index_key: str*, *data_path: str*, **args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseVectorIndexer*

add(*keys: List[Tuple[int, Any]], vectors: numpy.ndarray, weights: List[float], *args, **kwargs*)

normalize_score(*score: numpy.ndarray*, **args*, ***kwargs*) → numpy.ndarray

post_init()
Declare class attributes/members that can not be serialized in standard way

query()

size

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.indexer.vector.numpy module

class gnes.indexer.vector.numpy.**NumpyIndexer**(*num_bytes: int = None*, **args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseVectorIndexer*

add(*keys: List[Tuple[int, Any]], vectors: numpy.ndarray, weights: List[float], *args, **kwargs*)

query()

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

Submodules

gnes.indexer.base module

class gnes.indexer.base.**BaseIndexer**(**args*, ***kwargs*)
Bases: *gnes.base.TrainableBase*

add(*keys: Any, docs: Any, weights: List[float], *args, **kwargs*)

normalize_score(**args*, ***kwargs*)

query(*keys: Any*, **args*, ***kwargs*) → List[Any]

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.base.**BaseKeyIndexer**(**args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseIndexer*

add(*keys: List[Tuple[int, int]]*, *weights: List[float]*, **args*, ***kwargs*) → int

query(*keys: List[int]*, **args*, ***kwargs*) → List[Tuple[int, int, float]]

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.base.**BaseTextIndexer**(**args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseIndexer*

add(*keys: List[int]*, *docs: Any*, *weights: List[float]*, **args*, ***kwargs*)

query(*keys: List[int]*, **args*, ***kwargs*) → List[Any]

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.base.**BaseVectorIndexer**(**args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseIndexer*

add(*keys: List[Tuple[int, int]]*, *vectors: numpy.ndarray*, *weights: List[float]*, **args*, ***kwargs*)

query()

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.base.**JointIndexer**(**args*, ***kwargs*)
Bases: *gnes.base.CompositionalTrainableBase*

add(*keys: Any*, *docs: Any*, **args*, ***kwargs*) → None

components

query()

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.indexer.key_only module

class gnes.indexer.key_only.**DictKeyIndexer**(**args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseKeyIndexer*

add(*keys: List[Tuple[int, int]]*, *weights: List[float]*, **args*, ***kwargs*) → int

query(*keys: List[int]*, **args*, ***kwargs*) → List[Tuple[int, int, float]]

size

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.key_only.**ListKeyIndexer**(**args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseKeyIndexer*

add(*keys: List[Tuple[int, int]]*, *weights: List[float]*, **args*, ***kwargs*) → int

query(*keys: List[int], *args, **kwargs*) → List[Tuple[int, int, float]]

size

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.key_only.**ListNumpyKeyIndexer**(**args*, ***kwargs*)
Bases: *gnes.indexer.key_only.ListKeyIndexer*

add(**args*, ***kwargs*) → int

query(*keys: List[int], *args, **kwargs*) → List[Tuple[int, int, float]]

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.indexer.key_only.**NumpyKeyIndexer**(*buffer_size: int = 10000*, *col_size: int = 3*, **args*, ***kwargs*)
Bases: *gnes.indexer.base.BaseKeyIndexer*

add(*keys: List[Tuple[int, int]]*, *weights: List[float]*, **args*, ***kwargs*) → int

capacity

query(*keys: List[int], *args, **kwargs*) → List[Tuple[int, int, float]]

size

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.preprocessor package

Subpackages

gnes.preprocessor.audio package

Submodules

gnes.preprocessor.audio.audio_vanilla module

class gnes.preprocessor.audio.audio_vanilla.**AudioVanilla**(*audio_interval: int*, *sample_rate: int*, **args*, ***kwargs*)
Bases: *gnes.preprocessor.base.BaseAudioPreprocessor*

apply(*doc: gnes_pb2.Document*) → None

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.preprocessor.image package

Submodules

gnes.preprocessor.image.resize module

class gnes.preprocessor.image.resize.**ResizeChunkPreprocessor**(*target_width: int = 224*, *target_height: int = 224*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.image.resize.SizedPreprocessor*

apply(*doc: gnes_pb2.Document*) → None

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.preprocessor.image.resize.**SizedPreprocessor**(*target_width: int = 224*, *target_height: int = 224*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.base.BaseImagePreprocessor*

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.preprocessor.image.segmentation module

class gnes.preprocessor.image.segmentation.**SegmentPreprocessor**(*model_name: str*, *model_dir: str*, *_use_cuda: bool = False*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.image.resize.SizedPreprocessor*

apply(*doc: gnes_pb2.Document*)

post_init()
Declare class attributes/members that can not be serialized in standard way

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.preprocessor.image.sliding_window module

class gnes.preprocessor.image.sliding_window.**VanillaSlidingPreprocessor**(*window_size: int = 64*, *stride_height: int = 64*, *stride_wide: int = 64*, **args*, ***kwargs*)

Bases: gnes.preprocessor.image.sliding_window._SlidingPreprocessor

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.preprocessor.image.sliding_window.**WeightedSlidingPreprocessor**(*window_size: int = 64*, *stride_height: int = 64*, *stride_wide: int = 64*, **args*, ***kwargs*)

Bases: gnes.preprocessor.image.sliding_window._SlidingPreprocessor

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.preprocessor.text package

Submodules

gnes.preprocessor.text.split_punct module

class gnes.preprocessor.text.split_punct.**PunctSplitPreprocessor**(*deliminator: str = '[.!?]+'*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.base.BaseTextPreprocessor*

apply(*doc: gnes_pb2.Document*) → None

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.preprocessor.video package

Submodules

gnes.preprocessor.video.ffmpeg module

gnes.preprocessor.video.shotdetect module

Module contents

Submodules

gnes.preprocessor.base module

class gnes.preprocessor.base.**BaseAudioPreprocessor**(*uniform_doc_weight: bool = True*, **args*, ***kwargs*)
Bases: *gnes.preprocessor.base.BasePreprocessor*

doc_type = 4

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.preprocessor.base.**BaseImagePreprocessor**(*uniform_doc_weight: bool = True*, **args*, ***kwargs*)
Bases: *gnes.preprocessor.base.BasePreprocessor*

doc_type = 2

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.preprocessor.base.**BasePreprocessor**(*uniform_doc_weight: bool = True*, **args*, ***kwargs*)
Bases: *gnes.base.TrainableBase*

apply(*doc: gnes_pb2.Document*) → None

doc_type = 0

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.preprocessor.base.**BaseTextPreprocessor**(*uniform_doc_weight: bool = True*, **args*, ***kwargs*)
Bases: *gnes.preprocessor.base.BasePreprocessor*

doc_type = 1

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.preprocessor.base.**BaseVideoPreprocessor**(*uniform_doc_weight: bool = True*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.base.BasePreprocessor*

doc_type = 3

train(**args*, ***kwargs*)

Train the model, need to be overrided

class gnes.preprocessor.base.**PipelinePreprocessor**(**args*, ***kwargs*)

Bases: *gnes.base.CompositionalTrainableBase*

apply(*doc: gnes_pb2.Document*) → None

train(*data*, **args*, ***kwargs*)

Train the model, need to be overrided

class gnes.preprocessor.base.**RawChunkPreprocessor**(*uniform_doc_weight: bool = True*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.base.BasePreprocessor*

apply(*doc: gnes_pb2.Document*) → None

train(**args*, ***kwargs*)

Train the model, need to be overrided

class gnes.preprocessor.base.**UnaryPreprocessor**(*doc_type: int*, **args*, ***kwargs*)

Bases: *gnes.preprocessor.base.BasePreprocessor*

apply(*doc: gnes_pb2.Document*)

is_trained = True

raw_to_chunk(*chunk: gnes_pb2.Chunk*, *raw_bytes: bytes*)

train(**args*, ***kwargs*)

Train the model, need to be overrided

gnes.preprocessor.helper module

gnes.preprocessor.helper.**block_descriptor**(*image: numpy.ndarray*, *descriptor_fn: Callable*, *num_blocks: int = 3*) → numpy.ndarray

gnes.preprocessor.helper.**canny_edge**(*image: numpy.ndarray*, ***kwargs*) → numpy.ndarray

gnes.preprocessor.helper.**check_motion**(*prev_dists: List[float]*, *cur_dist: float*, *motion_threshold: float = 0.75*)

Returns a boolean value to decide if the peak is due to a motion

gnes.preprocessor.helper.**compare_descriptor**(*descriptor1: numpy.ndarray*, *descriptor2: numpy.ndarray*, *metric: str = 'chisqr'*) → float

gnes.preprocessor.helper.**compare_ecr**(*descriptors: List[np.ndarray]*, *dilate_rate: int = 5*, *neigh_avg: int = 2*) → List[float]

gnes.preprocessor.helper.**compute_descriptor**(*image: numpy.ndarray*, *method: str = 'rgb_histogram'*, ***kwargs*) → numpy.array

gnes.preprocessor.helper.**detect_peak_boundary**(*distances: List[float]*, *method: str = 'kmeans'*, ***kwargs*) → List[int]

gnes.preprocessor.helper.**get_all_subarea**(*img*)

gnes.preprocessor.helper.**get_audio**(*buffer_data*, *sample_rate*, *interval*, *duration*) → List[numpy.ndarray]

gnes.preprocessor.helper.**get_gif**(*images*, *fps=4*)

gnes.preprocessor.helper.**get_video_length**(*video_path*)

gnes.preprocessor.helper.**get_video_length_from_raw**(*buffer_data*)

gnes.preprocessor.helper.**hsv_histogram**(*image: numpy.ndarray*) → numpy.ndarray

gnes.preprocessor.helper.**kmeans_algo**(*distances: List[float], **kwargs*) → List[int]

gnes.preprocessor.helper.**motion_algo**(*distances: List[float], **kwargs*) → List[int]

gnes.preprocessor.helper.**phash_descriptor**(*image: numpy.ndarray*)

gnes.preprocessor.helper.**pyramid_descriptor**(*image: numpy.ndarray*, *descriptor_fn: Callable*, *max_level: int = 2*) → numpy.ndarray

gnes.preprocessor.helper.**rgb_histogram**(*image: numpy.ndarray*) → numpy.ndarray

gnes.preprocessor.helper.**split_mp4_random**(*video_path*, *avg_length*, *max_clip_second=10*)

gnes.preprocessor.helper.**split_video_frames**(*buffer_data: bytes*, *splitter: str = '__split__'*)

gnes.preprocessor.helper.**thre_algo**(*distances: List[float], **kwargs*) → List[int]

gnes.preprocessor.helper.**torch_transform**(*img*)

Module contents

gnes.proto package

Submodules

gnes.proto.gnes_pb2 module

gnes.proto.gnes_pb2_grpc module

class gnes.proto.gnes_pb2_grpc.**GnesRPCServicer**
Bases: object

Call(*request*, *context*)

Index(*request*, *context*)

Query(*request*, *context*)

StreamCall(*request_iterator*, *context*)

Train(*request*, *context*)
option (rpc_core.method_no_deadline) = true; option (rpc_core.service_default_deadline_ms) = 5000;

class gnes.proto.gnes_pb2_grpc.**GnesRPCStub**(*channel*)
Bases: object

Constructor.

Args: channel: A grpc.Channel.

gnes.proto.gnes_pb2_grpc.**add_GnesRPCServicer_to_server**(*servicer*, *server*)

Module contents

class gnes.proto.**RequestGenerator**
Bases: object

static index(*data: List[bytes], batch_size: int = 0, doc_type: int = 1, doc_id_start: int = 0, request_id_start: int = 0, random_doc_id: bool = False, *args, **kwargs*)

static query(*query: bytes*, *top_k: int*, *request_id_start: int = 0*, *doc_type: int = 1*, **args*, ***kwargs*)

static train(*data: List[bytes], batch_size: int = 0, doc_type: int = 1, doc_id_start: int = 0, request_id_start: int = 0, random_doc_id: bool = False, *args, **kwargs*)

gnes.proto.**send_message**(*sock: zmq.sugar.socket.Socket*, *msg: gnes_pb2.Message*, *timeout: int = -1*) → None

gnes.proto.**recv_message**(*sock: zmq.sugar.socket.Socket*, *timeout: int = -1*) → Optional[gnes_pb2.Message]

gnes.proto.**blob2array**(*blob: gnes_pb2.NdArray*) → numpy.ndarray
Convert a blob proto to an array.

gnes.proto.**array2blob**(*x: numpy.ndarray*) → gnes_pb2.NdArray
Converts a N-dimensional array to blob proto.

gnes.proto.**add_route**(*evlp: gnes_pb2.Envelope*, *name: str*)

gnes.router package

Subpackages

gnes.router.map package

Submodules

gnes.router.map.simple module

class gnes.router.map.simple.**DocBatchRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseMapRouter*

apply(*msg: gnes_pb2.Message*, **args*, ***kwargs*) → Generator
Modify the incoming message

Parameters **msg** – incoming message

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.router.map.simple.**PublishRouter**(*num_part: int*, **args*, ***kwargs*)
Bases: *gnes.router.base.BaseMapRouter*

apply(*msg: gnes_pb2.Message*, **args*, ***kwargs*) → Generator
Modify the incoming message

Parameters **msg** – incoming message

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.router.reduce package

Submodules

gnes.router.reduce.chunk module

class gnes.router.reduce.chunk.**ChunkToDocRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseReduceRouter*

apply(*msg: gnes_pb2.Message, accum_msgs: List[gnes_pb2.Message], *args, **kwargs*)
Modify the current message based on accumulated messages

Parameters

- **msg** – the current message
- **accum_msgs** – accumulated messages

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.router.reduce.chunk_sum module

class gnes.router.reduce.chunk_sum.**ChunkSumRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseReduceRouter*

apply(*msg: gnes_pb2.Message, accum_msgs: List[gnes_pb2.Message], *args, **kwargs*)
Modify the current message based on accumulated messages

Parameters

- **msg** – the current message
- **accum_msgs** – accumulated messages

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.router.reduce.concat module

class gnes.router.reduce.concat.**ConcatEmbedRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseReduceRouter*

apply(*msg: gnes_pb2.Message, accum_msgs: List[gnes_pb2.Message], *args, **kwargs*)
Modify the current message based on accumulated messages

Parameters

- **msg** – the current message
- **accum_msgs** – accumulated messages

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.router.reduce.doc_sum module

class gnes.router.reduce.doc_sum.**DocSumRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseReduceRouter*

apply(*msg: gnes_pb2.Message, accum_msgs: List[gnes_pb2.Message], *args, **kwargs*)
Modify the current message based on accumulated messages

Parameters

- **msg** – the current message
- **accum_msgs** – accumulated messages

train(**args*, ***kwargs*)
Train the model, need to be overrided

gnes.router.reduce.document module

class gnes.router.reduce.document.**DocFillRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseReduceRouter*

apply(*msg: gnes_pb2.Message, accum_msgs: List[gnes_pb2.Message], *args, **kwargs*)
Modify the current message based on accumulated messages

Parameters

- **msg** – the current message
- **accum_msgs** – accumulated messages

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

Submodules

gnes.router.base module

class gnes.router.base.**BaseMapRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseRouter*

apply(*msg: gnes_pb2.Message*, **args*, ***kwargs*) → Generator
Modify the incoming message

Parameters **msg** – incoming message

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.router.base.**BaseReduceRouter**(**args*, ***kwargs*)
Bases: *gnes.router.base.BaseRouter*

apply(*msg: gnes_pb2.Message, accum_msgs: List[gnes_pb2.Message], *args, **kwargs*) → None
Modify the current message based on accumulated messages

Parameters

- **msg** – the current message
- **accum_msgs** – accumulated messages

train(**args*, ***kwargs*)
Train the model, need to be overrided

class gnes.router.base.**BaseRouter**(**args*, ***kwargs*)
Bases: *gnes.base.TrainableBase*

Base class for the router. Inherit from this class to create a new router.

Router forwards messages between services. Essentially, it receives a 'gnes_pb2.Message' and call *apply()* method on it.

apply(*msg: gnes_pb2.Message*, **args*, ***kwargs*)
Modify the incoming message

Parameters **msg** – incoming message

train(**args*, ***kwargs*)
Train the model, need to be overrided

Module contents

gnes.service package

Submodules

gnes.service.base module

class gnes.service.base.**BaseService**(*args*)
Bases: object

close()

default_host = '0.0.0.0'

dump()

handler = <gnes.service.base.MessageHandler object>

load_model(*base_class: Type[gnes.base.TrainableBase]*) → T

message_handler(*msg: gnes_pb2.Message*, *out_sck*, *ctrl_sck*)

post_init()

run()

status

class gnes.service.base.**BetterEnum**
Bases: enum.Enum

An enumeration.

from_string = <bound method BetterEnum.from_string of <enum 'BetterEnum'>>

exception gnes.service.base.**BlockMessage**
Bases: Exception

exception gnes.service.base.**ComponentNotLoad**
Bases: Exception

class gnes.service.base.**ConcurrentService**
Bases: type

exception gnes.service.base.**EventLoopEnd**
Bases: Exception

class gnes.service.base.**MessageHandler**(*mh: Optional[gnes.service.base.MessageHandler] = None*)
Bases: object

register()

serve(*msg: gnes_pb2.Message*)

class gnes.service.base.**ParallelType**
Bases: *gnes.service.base.BetterEnum*

An enumeration.

PUB_BLOCK = 2

PUB_NONBLOCK = 3

PUSH_BLOCK = 0

PUSH_NONBLOCK = 1

is_block

is_push

class gnes.service.base.**ReduceOp**
Bases: *gnes.service.base.BetterEnum*

An enumeration.

ALWAYS_ONE = 1

CONCAT = 0

exception gnes.service.base.**ServiceError**
Bases: Exception

class gnes.service.base.**ServiceManager**(*service_cls*, *args*)
Bases: object

join()

class gnes.service.base.**SocketType**
Bases: *gnes.service.base.BetterEnum*

An enumeration.

PAIR_BIND = 8

PAIR_CONNECT = 9

PUB_BIND = 6

PUB_CONNECT = 7

PULL_BIND = 0

PULL_CONNECT = 1

PUSH_BIND = 2

PUSH_CONNECT = 3

SUB_BIND = 4

SUB_CONNECT = 5

is_bind

gnes.service.base.**build_socket**(*ctx: zmq.sugar.context.Context*, *host: str*, *port: int*, *socket_type: gnes.service.base.SocketType*, *identity: Optional[str] = None*) → Tuple[zmq.sugar.socket.Socket, str]

gnes.service.base.**send_ctrl_message**(*address: str*, *msg: gnes_pb2.Message*, *timeout: int*)

gnes.service.encoder module

class gnes.service.encoder.**EncoderService**(*args*)

Bases: *gnes.service.base.BaseService*

static get_chunks_from_docs(*docs: Union[List[gnes_pb2.Document], gnes_pb2.Document]*) → List

handler = <gnes.service.base.MessageHandler object>

post_init()

gnes.service.frontend module

class gnes.service.frontend.**FrontendService**(*args*)

Bases: object

gnes.service.grpc module

class gnes.service.grpc.**GRPCService**(*args*)

Bases: *gnes.service.base.BaseService*

close()

handler = <gnes.service.base.MessageHandler object>

post_init()

gnes.service.indexer module

class gnes.service.indexer.**IndexerService**(*args*)

Bases: *gnes.service.base.BaseService*

handler = <gnes.service.base.MessageHandler object>

post_init()

gnes.service.preprocessor module

class gnes.service.preprocessor.**PreprocessorService**(*args*)
Bases: *gnes.service.base.BaseService*

handler = <gnes.service.base.MessageHandler object>

post_init()

gnes.service.router module

class gnes.service.router.**RouterService**(*args*)
Bases: *gnes.service.base.BaseService*

handler = <gnes.service.base.MessageHandler object>

post_init()

Module contents

1.2.2 Submodules

gnes.component module

gnes.helper module

gnes.helper.**get_sys_info**()

gnes.helper.**get_optimal_sample_size**(*x*)

gnes.helper.**get_perm**(*L*, *m*)

gnes.helper.**time_profile**(*func*)

gnes.helper.**set_logger**(*context*, *verbose=False*)

gnes.helper.**batch_iterator**(*data: Union[Iterator[Any], List[Any], numpy.ndarray], batch_size: int, axis: int = 0*) → Iterator[Any]

gnes.helper.**batching**(*func: Callable[Any, numpy.ndarray] = None*, **, batch_size: Union[int, Callable] = None, num_batch=None, iter_axis: int = 0, concat_axis: int = 0, chunk_dim=-1*)

gnes.helper.**load_contrib_module**()

gnes.helper.**parse_arg**(*v: str*)

gnes.helper.**profiling**(*func*)

class gnes.helper.**FileLock**(*lock_file: str = 'LOCK'*)
Bases: object

Implements the Posix based file locking (Linux, Ubuntu, MacOS, etc.)

acquire()

is_locked

release()

gnes.helper.**train_required**(*func*)

gnes.helper.**get_first_available_gpu**()

class gnes.helper.**PathImporter**
Bases: object

static add_modules(**paths*)

1.2.3 Module contents

1.3 Contributing to GNES

Thanks for your interest in contributing! GNES always welcome the contribution from the open-source community, individual committers and other partners. Without you, GNES can't be successful.

Currently there are three major directions of contribution:

- **Porting state-of-the-art models to GNES**. This includes new preprocessing algorithms, new DNN networks for encoding, and new high-performance index. Believe me, it is super easy to wrap an algorithm and use it in GNES. Checkout this example.
- **Adding tutorial and learning experience**. What is good and what can be improved? If you apply GNES in your domain, whether it's about NLP or CV, whether it's a blog post or a Reddit/Twitter thread, we are always eager to hear your thoughts.
- **Completing the user experience of other programming languages**. GNES offers a generic interface with gRPC and protobuf, therefore it is easy to add an interface for other languages, e.g. Java, C, Go.

1.3.1 Table of Content

1.3.2 Commit Message Naming

To help everyone with understanding the commit history of GNES, we employ `commitlint` in the CI pipeline to enforce the commit styles. Specifically, our convention is:

```
type(scope?): subject
```

where `type` is one of the following:

- build
- ci
- chore
- docs
- feat
- fix

- perf
- refactor
- revert
- style
- test

`scope` is optional, represents the module your commit working on.

`subject` explains the commit.

As an example, a commit that implements a new encoder should be phrased as:

```
feat(encoder): add new inceptionV3 as image encoder
```

1.3.3 Merging Process

A pull request has to meet the following conditions to be merged into master:

- Coding style check (PEP8, via Codacy)
- Commit style check (in CI pipeline via Drone.io)
- Unit tests (via Drone.io)
- Review and approval from a GNES team member.

After the merging is triggered, the build will be delivered to the followings:

- **Docker Hub**: `gnes:latest` will be updated.
- **Tencent Container Service**: `gnes:latest` will be updated.
- **ReadTheDoc**: `latest` will be updated.

Note that merging into master does not mean an official releasing. For the releasing process, please refer to the next section.

1.3.4 Release Process

A new release is scheduled on every Friday (triggered and approved by Han Xiao) summarizing all new commits since the last release. The release will increment the third (revision) part of the version number, i.e. from `0.0.24` to `0.0.25`.

After a release is triggered, the build will be delivered to the followings:

- **Docker Hub**: a new image with the release version tag will be created, `gnes:latest` will be updated.
- **Tencent Container Service**: a new image with the release version tag will be created, `gnes:latest` will be updated.
- **PyPi Package**: a new version of Python package is uploaded to Pypi, allowing one to `pip install -U gnes`
- **ReadTheDoc**: a new version of the document will be built, `latest` will be updated and the old version will be achieved.

Meanwhile, a new pull request containing the updated CHANGELOG and the new version number will be made automatically, pending for review and merge.

Major and minor version increments

- MAJOR version when GNES make incompatible API changes;
- MINOR version when GNES add functionality in a backwards-compatible manner.

The decision of incrementing major and minor version, i.e. from `0.0.0` to `0.1.0` or from `1.0.0` to `2.0.0`, is made by the GNES team.

1.3.5 Testing Locally

The best way to test GNES is using a Docker container, in which you don't have to worry about the dependencies.

We provide a public Docker image `gnes/ci-base`, which contains the required dependencies and some pretrained models used in our continuous integration pipeline.

You can find the image at here or pull the image via:

```
docker pull gnes/ci-base
```

To test GNES inside this image, you may run

```
docker run --network=host --rm --entrypoint "/bin/bash" -it gnes/ci-base

# now you are inside the 'gnes/ci-base' container
# first sync your local modification, then
pip install -e .[all]
python -m unittest tests/*.py
```

1.4 Release Note (`v0.0.34`)

Release time: 2019-08-23 19:00:27

We'd like to thank all contributors for this new release! In particular, hanhxiao,

1.4.1 Other Improvements

- [`79a8effd`] **- changelog**: update change log to v0.0.33 (*hanhxiao*)

1.5 Release Note (`v0.0.34`)

Release time: 2019-08-23 18:44:34

We'd like to thank all contributors for this new release! In particular, hanhxiao,

1.5.1 Other Improvements

- [`79a8effd`] **- changelog**: update change log to v0.0.33 (*hanhxiao*)

1.6 Release Note (`v0.0.33`)

Release time: 2019-08-23 18:34:28

We'd like to thank all contributors for this new release! In particular, Jem, hanhxiao, felix, raccoonliukai,

1.6.1 New Features

- [`9d488e3f`] - **client**: add progress bar and speed metric to cli (*hanhxiao*)
- [`829d148c`] - **scale_video**: scale video use ffmpeg (*felix*)
- [`bc2e441d`] - **compose**: add minimum http server without flask dep (*hanhxiao*)
- [`d420f348`] - **video preprocessor**: add edge detect for shotdetect (*raccoonliukai*)

1.6.2 Bug fixes

- [`6cfbda9d`] - **preprocessor**: move dependency into function (*Jem*)
- [`0e88b77a`] - **frontend**: fix request_id zero is none (*hanhxiao*)
- [`ca28ecb9`] - **video preprocessor**: use rgb as standard color (*raccoonliukai*)
- [`5b5feb0b`] - **video preprocessor**: use dict update (*raccoonliukai*)
- [`47721b1c`] - **video preprocessor**: remove custom canny threshold (*raccoonliukai*)
- [`16aaa777`] - **video preprocessor**: modify inaccurate names (*raccoonliukai*)
- [`dfb54b62`] - **video preprocessor**: Remove incorrect comments (*raccoonliukai*)

1.6.3 Code Refactoring

- [`3d63fac6`] - **proto**: request_id is now an integer (*hanhxiao*)
- [`4497d765`] - **shotdetector**: use updated ffmpeg api to capture frames from videos (*felix*)
- [`dbc06a85`] - **ffmpeg**: refactor ffmpeg to read frames, vides and gif (*felix*)
- [`a7b12cb6`] - **preprocessor**: add gif chunk prep (*Jem*)
- [`559a9971`] - **compose**: unify flask and http handler (*hanhxiao*)

1.6.4 Documentation

- [`a2801d5c`] - link gnes hub tutorial to readme (*hanhxiao*)

1.6.5 Other Improvements

- [`02f70a03`] - fix bug (*felix*)
- [`c970bec3`] - **changelog**: update change log to v0.0.32 (*hanhxiao*)

1.7 Release Note (`v0.0.32`)

Release time: 2019-08-21 17:23:13

We'd like to thank all contributors for this new release! In particular, hanhxiao, Han Xiao, Jem,

1.7.1 New Features

- [38567b00] - **indexer**: add preprocessor and lvdb for storing gif (*Jem*)
- [35465e85] - **base**: later import module now override the earlier ones (*hanhxiao*)

1.7.2 Bug fixes

- [5c2b60a4] - remove target_image_size (*hanhxiao*)
- [944b8c09] - **ci**: fix unit tests for modules (*hanhxiao*)

1.7.3 Code Refactoring

- [5f1ca000] - fixing the imports of all base module (*hanhxiao*)
- [0d1bd4e2] - **preprocessor**: remove unnecessary init (*Han Xiao*)

1.7.4 Unit Test and CICD

- [3820db6a] - **encoder**: rename BasePytorchEncoder to TorchvisionEncoder (*hanhxiao*)

1.7.5 Other Improvements

- [3147a1d5] - **changelog**: update change log to v0.0.31 (*hanhxiao*)

1.8 Release Note (`v0.0.31`)

Release time: 2019-08-20 14:01:04

We'd like to thank all contributors for this new release! In particular, hanhxiao,

1.8.1 New Features

- [f7beae7b] - **cli**: add py_path in parser to load external modules (*hanhxiao*)

1.8.2 Other Improvements

- [ec1eb787] - **release**: fix duplicate release notes (*hanhxiao*)
- [447756d5] - **changelog**: update change log to v0.0.30 (*hanhxiao*)

1.9 Release Note (v0.0.30)

Release time: 2019-08-19 14:13:03

We'd like to thank all contributors for this new release! In particular, hanhxiao,

1.9.1 New Features

- [7b5cc86a] **- contrib**: no need to give module name in advance (*hanhxiao*)

1.9.2 Bug fixes

- [5f69c781] **- contrib**: allowing dump for contribued module (*hanhxiao*)

1.9.3 Other Improvements

- [565ef569] **- changelog**: update change log to v0.0.29 (*hanhxiao*)

1.10 Release Note (v0.0.29)

Release time: 2019-08-16 15:40:31

We'd like to thank all contributors for this new release! In particular, hanhxiao, Jem,

1.10.1 Bug fixes

- [2f905168] **- setup**: fix encoding problem in setup.py (*hanhxiao*)
- [469bc51d] **- ci**: fix cd pipeline (*hanhxiao*)

1.10.2 Code Refactoring

- [66d020bd] **- base**: component renamed to components (*hanhxiao*)
- [3a2b85b6] **- proto**: assign doc id in request generator (*Jem*)

1.10.3 Documentation

- [b854c697] **- readme**: fix description on images (*hanhxiao*)

1.10.4 Unit Test and CICD

- [f1658c92] **- docker**: docker image tag-alpine as default tag (*hanhxiao*)
- [8885512f] **- docker**: clean up the space after docker build (*hanhxiao*)

1.10.5 Other Improvements

- [00ca6919] - **changelog**: update change log to v0.0.28 (*hanhxiao*)

1.11 Release Note (v0.0.28)

Release time: 2019-08-14 20:54:26

We'd like to thank all contributors for this new release! In particular, hanhxiao, Jem, raccoonliukai, Larry Yan,

1.11.1 New Features

- [0133905c] - **client**: add a client for benchmarking and testing (*hanhxiao*)
- [732f2e64] - **encoder**: add pytorch transformers support in text encoder (*raccoonliukai*)
- [6aab48c8] - **docker**: add buster image with minimum dependencies (*hanhxiao*)
- [da1bbc0d] - **docker**: add alpine image with minimum dependencies (*hanhxiao*)

1.11.2 Bug fixes

- [315bd16a] - **doc sum router**: use meta info instead of doc id to do doc sum (*Jem*)
- [c9e92722] - **encoder**: use offline model in ci-base for pytorch transformer (*raccoonliukai*)
- [d7b42d39] - **setup**: remove unused dependencies (*hanhxiao*)
- [5b8acf7c] - **test**: fix routes assert in tests (*hanhxiao*)
- [5fedf6df] - **encoder**: fix unused variable (*raccoonliukai*)
- [df616463] - **cli**: remove unnecessary argument (*hanhxiao*)
- [fd76aa79] - **request_generator**: send index request in index mode (*Jem*)
- [64163cb1] - **batching**: enable to process three dimension output in batching (*Jem*)
- [415456d6] - **preprocessor**: fix bug (*Larry Yan*)
- [c150ad59] - **preprocessor**: modify ffmpeg video pre add video cutting method (*Larry Yan*)
- [b0f22d04] - **audio preprocessor**: filter audio with zero length (*Jem*)
- [d1cfa539] - **preprocessor**: modify ffmpeg video preprocessor (*Larry Yan*)

1.11.3 Documentation

- [e11a920e] - **readme**: add image explain to readme (*hanhxiao*)

1.11.4 Unit Test and CICD

- [a8700801] - **drone**: add self hosted drone (*hanhxiao*)
- [079d0a1a] - **docker**: move docker-build to a more controllable cd process (*hanhxiao*)

1.11.5 Other Improvements

- [5257259f] - add kai liu to core maintainers (*hanhxiao*)
- [8d318204] - **changelog**: update change log to v0.0.27 (*hanhxiao*)

1.12 Release Note (v0.0.27)

Release time: 2019-08-09 19:51:57

We'd like to thank all contributors for this new release! In particular, hanhxiao, Jem, Larry Yan, raccoonliu, Han Xiao, raccoonliukai,

1.12.1 New Features

- [55126f2b] - **grpc**: add a general purpose grpc service (*hanhxiao*)
- [23c6e68a] - **reduce router**: add chunk and doc reduce routers for audio (*Jem*)
- [6d3d2b4c] - **cli**: use ServiceManager as default service runner (*hanhxiao*)
- [ccfd474a] - **service**: add ServiceManager and enable parallel services in one container (*hanhxiao*)
- [63f9173f] - **service**: enabling the choose of thread or process as the backend (*hanhxiao*)
- [2647b848] - **audio**: add preprocess and mfcc encoder for audio (*Jem*)
- [208e1937] - **audio**: add preprocess and mfcc encoder for audio, update protobuf (*Jem*)
- [77a2ea42] - **parser**: improve yaml_path parsing (*hanhxiao*)
- [762535ca] - **vlad**: add vlad and enable multiple chunks and frames (*Jem*)
- [64e948d4] - **encoder**: add onnxruntime for image encoder (*raccoonliukai*)
- [f03e6fc2] - **encoder**: add onnxruntime suport for image encoder (*raccoonliukai*)

1.12.2 Bug fixes

- [5ae46d61] - **composer**: rename grpcfrontend to frontend (*hanhxiao*)
- [4cb83383] - **audio**: restrict max length for mfcc encoding (*Jem*)
- [e516646f] - **grpc**: add max_message_size to the argparser (*hanhxiao*)
- [0493e6fc] - **encoder**: fix netvlad (*Larry Yan*)
- [e773aa33] - **service manager**: fix nonetype for service manager (*Jem*)
- [d5d15d7f] - **compose**: fix a bug in doc_reduce_test (*hanhxiao*)
- [6856cb0a] - **compose**: copy args on every request (*hanhxiao*)
- [f80e8c03] - **cli**: set default num_part is None (*hanhxiao*)
- [7031fe20] - **preprocessor**: add random sampling to ffmpeg (*Larry Yan*)
- [fd37e6d9] - **encoder**: fix bug caused by batching in inception_mixture (*Larry Yan*)
- [2191b27b] - **composer**: fix yaml generation (*hanhxiao*)
- [e5fefcee] - **encoder**: fix batching in encoder (*hanhxiao*)

- [e35e3b3c] - **composer**: fix composer router generation logic (*hanhxiao*)
- [7300e055] - **preprocessor**: quanlity improvement (*Larry Yan*)
- [47efaba4] - **unittest**: fix unittest of video preprocessor 2 (*Larry Yan*)
- [a6efb4af] - **unittest**: fix unittest of video preprocessor (*Larry Yan*)
- [dd1216bb] - **unittest**: fix unittest for video processor (*Larry Yan*)
- [8e6dc4c6] - **encoder**: add func for preprocessor (*Larry Yan*)
- [2b21dc5a] - **encoder**: fix unused import and variable (*raccoonliu*)
- [fd576915] - **test**: fix import (*Han Xiao*)
- [a0fdad36] - **test**: fix broken code (*Han Xiao*)
- [8ca07a74] - **test**: fix img_process_for_test (*Han Xiao*)
- [7c16fb8b] - **preprocessor**: fix bug in ffmpeg.py and add more func to helper (*Larry Yan*)
- [e6a37119] - **preprocessor**: fix bug in params in ffmepg (*Larry Yan*)
- [f8d2abe5] - **preprocessor**: fix bug in ffmpeg (*Larry Yan*)
- [67610f86] - **preprocessor**: add more method for cutting video (*Larry Yan*)

1.12.3 Code Refactoring

- [8516096d] - **grpc**: moving zmqclient to client module (*hanhxiao*)
- [5e3409e1] - **grpc**: hide private class inside gRPCfrontend (*hanhxiao*)
- [6407cc8d] - **yaml**: remove useless default yaml config (*hanhxiao*)
- [c1e406ae] - **onnx**: move batch_size to class attribute (*Han Xiao*)

1.12.4 Unit Test and CICD

- [5503dbe7] - skip joint indexer test as it is not even used (*hanhxiao*)
- [8ab101ca] - add mergify for auto merging (*hanhxiao*)
- [203d1697] - **chore**: exclude chore job from ci pipeline (*hanhxiao*)
- [24f9fd1c] - fix yaml_path missing in the test (*hanhxiao*)
- [23a83a40] - simplify yaml naming (*hanhxiao*)

1.12.5 Other Improvements

- [e8e3b9b9] - **changelog**: update change log to v0.0.26 (*hanhxiao*)

1.13 Release Note (`v0.0.26`)

Release time: 2019-08-02 18:18:45

We'd like to thank all contributors for this new release! In particular, hanhxiao, Jem, Larry Yan,

1.13.1 New Features

- [d0b2ef0b] - **composer**: more interaction for gnes board (*hanhxiao*)
- [9c33dc66] - **router**: allow consecutive mapping and reducing ops (*hanhxiao*)

1.13.2 Bug fixes

- [fc5026da] - **board**: improve gnes board 500 message (*hanhxiao*)
- [823bdeda] - **test**: fix grpc gentle shutdown (*hanhxiao*)
- [f6a801f7] - **test**: fix preprocessor building for image test (*hanhxiao*)
- [50fdc041] - **base**: fix ref to CompositionalTrainableBase (*hanhxiao*)
- [54a931c7] - **test**: fix test images by removing mac stuff (*hanhxiao*)
- [14cdfabe] - **sliding window**: fix the boundary (*Jem*)
- [46b5c94e] - **encoder**: fix name for video encoder (*Larry Yan*)
- [15eb50b4] - **encoder**: fix params in basevideo encoder (*Larry Yan*)
- [5b0fe7c6] - **preprocessor**: fix FFmpegVideoSegmentor (*Larry Yan*)
- [d6a46fa6] - **encoder**: fix import path for mixture encoder (*Larry Yan*)
- [17779676] - **encoder**: fix mixture encoder (*Larry Yan*)
- [95f03c56] - **encoder**: fix bug in video mixture encoder (*Larry Yan*)
- [3fdf1c06] - **encoder**: fix mixture (*Larry Yan*)
- [67991533] - **encoder**: add netvlad and netfv register class (*Larry Yan*)
- [92500f0f] - **encoder**: add netvlad and netfv (*Larry Yan*)

1.13.3 Code Refactoring

- [c430ef64] - **base**: better batch_size control (*hanhxiao*)
- [58217d8c] - **base**: moving is_trained to class attribute (*hanhxiao*)
- [7126d496] - **preprocessor**: separate resize logic from the unary preprocessor (*hanhxiao*)
- [52f87c7f] - **base**: make pipelineencoder more general and allow pipelinepreprocessor (*hanhxiao*)

1.13.4 Documentation

- [3ab3723e] - **tutorial**: fix image and code layout (*hanhxiao*)

1.13.5 Other Improvements

- [635ba37f] - **changelog**: update change log to v0.0.25 (*hanhxiao*)

1.14 Release Note (`v0.0.25`)

Release time: 2019-07-26 19:45:21

We'd like to thank all contributors for this new release! In particular, hanhxiao, felix, Larry Yan, Jem, Han Xiao, Felix,

1.14.1 New Features

- [66aec9c9] - **grpc**: add StreamCall and decouple send and receive (*hanhxiao*)
- [5697441b] - **indexer**: consider offset relevance at query time (*Jem*)
- [04c9c745] - **image preprocessor**: calculate offsetnd for each chunk (*Jem*)
- [b34a765a] - **compose**: add interactive mode of GNES board using Flask (*hanhxiao*)
- [5876c15e] - **base**: support loading external modules from py and yaml (*hanhxiao*)

1.14.2 Bug fixes

- [a20672d3] - **preprocessor**: add logging in helper module (*felix*)
- [f9500c1f] - **protobuffer**: add doc_type as func argument in RequestGenerator (*felix*)
- [1c3bb01a] - **service**: fix bug in doc_type name in indexer service (*Larry Yan*)
- [d834f578] - **service**: add doc type to req generator (*Larry Yan*)
- [80e234e1] - **service**: fix bug in req Generator add doc_type (*Larry Yan*)
- [5743e258] - **indexer**: fix bug in indexer service (*Larry Yan*)
- [11dde2bf] - **encoder**: fix bug in tf inception (*Larry Yan*)
- [ded92c57] - **indexer**: fix bug for indexer service dealing with empty doc (*Larry Yan*)
- [1dff06f1] - **encoder**: fix bug for encoder service dealing with empty doc (*Larry Yan*)
- [7e43d5a2] - **preprocessor**: fix ffmpeg to deal with broken image (*Larry Yan*)
- [83ebaced] - **preprocessor**: move import imagehash to inside (*hanhxiao*)
- [7c669a70] - **test**: rename the yaml test file (*hanhxiao*)
- [2cc26342] - **compose**: change textarea font to monospace (*hanhxiao*)
- [e644e391] - **encoder**: fix gpu limitation in inception (*Larry Yan*)
- [89d8b70c] - **grpc**: fix bug in RequestGenerator query (*Larry Yan*)
- [c52c2cc6] - **base**: fix gnes_config mixed in kwargs (*hanhxiao*)
- [68c15fac] - **base**: fix redundant warning in pipeline encoder (*hanhxiao*)
- [aadeeefb] - **composer**: fix composer state machine (*hanhxiao*)
- [c0bffe6c] - **indexer**: normalize weight (*Jem*)
- [2c696483] - **indexer**: fix weight in indexer call (*Larry Yan*)
- [139a02d9] - **compose**: fix compose bug of pub-sub rule, duplicate yaml_path (*hanhxiao*)
- [649ed131] - **encoder**: add normalize option in cvae encoder (*Larry Yan*)

- [eb487799] - **encoder**: fix tf scope error in cvae encoder (*Larry Yan*)
- [ab6c88cc] - **encoder**: fix error in cvae encoder (*Larry Yan*)
- [a4b883ac] - **indexer**: add drop raw bytes option to leveldb (*Larry Yan*)
- [4b52bcba] - **grpc**: fix grpc plugin path (*Larry Yan*)
- [d3fbbcac] - **weighting**: add simple normalization to chunk search (*Jem*)
- [08a9a4e3] - **grpc**: fix grpc service (*Larry Yan*)
- [6e6bbf83] - **grpc**: add auto-gen grpc code (*Larry Yan*)
- [b89d8fa2] - **grpc**: add stream index and train in proto (*Larry Yan*)
- [15cd7e58] - **base**: fix dump and load on compositional encoder (*hanhxiao*)
- [bab48919] - **encoder**: fix tf inception (*Larry Yan*)
- [973672ef] - **encoder**: fix bug for encoder bin load (*Larry Yan*)
- [1bef3971] - **setup**: fix setup script (*hanhxiao*)
- [67fb5766] - **compose**: fix argparser (*hanhxiao*)
- [63c4515f] - **compose**: accept parser argument only (*hanhxiao*)
- [887d89cc] - **release**: ask BOT_URL before releasing (*hanhxiao*)

1.14.3 Code Refactoring

- [9973f600] - **preprocessor**: rename singleton to unary (*hanhxiao*)
- [a1a2b020] - **proto**: refactor request stream call (*hanhxiao*)

1.14.4 Documentation

- [c853e3da] - **tutorial**: fix svg size (*hanhxiao*)
- [04cccdcd] - **tutorial**: fix svg path (*hanhxiao*)
- [8927cd4f] - **tutorial**: add yaml explain (*hanhxiao*)
- [5b52ce4c] - fix doc path (*hanhxiao*)
- [45751e1f] - **readme**: add quick start for readme (*hanhxiao*)
- [73891ecc] - **readme**: add install guide to readme and contribution guide (*hanhxiao*)

1.14.5 Unit Test and CICD

- [6ff3079b] - **unittest**: skip all os environ test (*hanhxiao*)
- [816fa043] - **unittest**: skip blocked test (*hanhxiao*)
- [79a9c106] - **unittest**: run test in verbose mode (*hanhxiao*)
- [83276f90] - **torchvision**: install torchvision dependency to enable tests (*hanhxiao*)
- [499682ce] - **base**: add unit test for load a dumped pipeline from yaml (*hanhxiao*)
- [26a7ad18] - **composer**: add unit test for flask (*hanhxiao*)

- [87ec1fd2] - **base**: move module delete to teardown (*hanhxiao*)
- [479b183d] - **compose**: skip unit test (*hanhxiao*)

1.14.6 Other Improvements

- [c30f39cc] - ... (*felix*)
- [2d5654c0] - **license**: add license (*hanhxiao*)
- [d3347910] - reformat code and optimize import (*hanhxiao*)
- [71491ffb] - **changelog**: update change log to v0.0.24 (*hanhxiao*)

1.15 Release Note (v0.0.24)

Release time: 2019-07-19 18:18:46

We'd like to thank all contributors for this new release! In particular, hanhxiao, Jem, Larry Yan, felix,

1.15.1 New Features

- [9f6c0524] - **fasterrcnn**: add the original image to chunk list (*Jem*)
- [abb0841c] - **encoder**: add convolution variational autoencoder (*Larry Yan*)

1.15.2 Bug fixes

- [1b526832] - **base**: fix dump yaml kwargs (*hanhxiao*)
- [086f3cea] - **base**: fix ump instance (*hanhxiao*)
- [12dfde42] - **base**: move name setting to trainable base (*hanhxiao*)
- [16f1a497] - **base**: move set config to metaclass (*hanhxiao*)
- [b97acd6c] - **base**: fix duplicate warning (*hanhxiao*)
- [991e4425] - **base**: fix duplicate load and init from yaml (*hanhxiao*)
- [69a486e5] - **compose**: fix import (*hanhxiao*)
- [4977aa3c] - **vector indexer**: reorder relevance and chunk weight (*Jem*)
- [2448411d] - **encoder**: modify CVAE (*Larry Yan*)
- [b4bf0bf8] - **indexer**: add path check for dir and file (*hanhxiao*)
- [92f36c33] - **fasterrcnn**: handle imgs with 0 chunk (*Jem*)
- [a1329913] - **fasterrcnn**: fix bug for gpu (*Jem*)
- [38eca0ce] - **grpc**: change grpc client message size limit (*felix*)
- [3836020a] - **preprocessor**: fix preprocessor service handler function name error (*felix*)
- [599a3c3d] - **compose**: fix composer logic (*hanhxiao*)
- [7f3b2fb5] - **release**: fix git tag version (*hanhxiao*)

1.15.3 Code Refactoring

- [9bbb3c05] - **compose**: move compose template to resources (*hanhxiao*)
- [a4e153d7] - **base**: remove dump path and reorganize work dir (*hanhxiao*)

1.15.4 Unit Test and CICD

- [e088ea9c] - **drone**: turn off profiling in ci (*hanhxiao*)
- [33a570b9] - **drone**: remove pylint for faster ci (*hanhxiao*)
- [51eafac7] - **indexer**: fix data path in unit test (*hanhxiao*)

1.15.5 Other Improvements

- [43ef4108] - **git**: add tmp to ignore (*hanhxiao*)
- [44b1a0c9] - fix unittest (*felix*)
- [984a9a2d] - **changelog**: update change log to v0.0.23 (*hanhxiao*)

1.16 Release Note (v0.0.23)

Release time: 2019-07-17 18:28:08

We'd like to thank all contributors for this new release! In particular, hanhxiao, Jem, felix, Larry Yan, Han Xiao,

1.16.1 New Features

- [cb4d9cf2] - **release**: add auto release and keep change log (*hanhxiao*)
- [c667d874] - **image_preprocessor**: add fasterRCNN (*Jem*)
- [a6c2975b] - **composer**: improve the gnes board with cards (*hanhxiao*)
- [6ec4233d] - **composer**: add swarm and bash generator (*hanhxiao*)
- [08aa30f4] - **composer**: add shell script generator (*hanhxiao*)
- [033a4b9c] - **composer**: add composer and mermaid renderer (*hanhxiao*)

1.16.2 Bug fixes

- [2b7c3f18] - **compose**: resolve unclosed file warning (*hanhxiao*)
- [8030feb2] - **compose**: fix router logic in compose (*hanhxiao*)
- [736f6053] - **gnesboard**: fix cdn (*hanhxiao*)
- [fb07ff02] - **doc_reducer_router**: fix reduce error (*felix*)
- [a7236308] - **image encoder**: define use_cuda variable via args (*felix*)
- [cba5e190] - **image_encoder**: enable batching encoding (*felix*)
- [3423ec83] - **composer**: add compose api to api.py (*hanhxiao*)

- [70ba3fca] - **composer**: in bash mode always run job in background (*hanhxiao*)
- [054981ce] - **composer**: fix gnes board naming (*hanhxiao*)
- [743ec3b0] - **composer**: fix unit test and add tear down (*hanhxiao*)
- [64aef413] - **composer**: fix styling according to codacy (*hanhxiao*)
- [dca4b03b] - **service**: fix bug grpc (*Larry Yan*)
- [09e68da2] - **service**: fix grpc server size limit (*Larry Yan*)
- [3da8da19] - **encoder**: rm un-used import in inception (*Larry Yan*)
- [8780a4da] - bugs for integrated test (*Jem*)
- [38fff782] - **preprocessor**: move cv2 dep to pic_weight (*Han Xiao*)
- [37155bba] - **preprocessor-video**: move sklearn dep to apply (*Han Xiao*)
- [1f6a06a2] - **encoder**: rm tf inception unittest (*Larry Yan*)
- [eaffbbff] - **encoder**: register tf inception in **init** (*Larry Yan*)
- [d0099b79] - **encoder**: add necessary code from tf (*Larry Yan*)
- [b480774a] - **encoder**: add inception tf (*Larry Yan*)

1.16.3 Documentation

- [54276c6a] - **readme**: improve readme image and structure (*hanhxiao*)

1.16.4 Unit Test and CICD

- [1dcfdfa7] - **docker-image**: optimze docker file (*felix*)
- [bda562d1] - **drone**: auto release with cron job (*hanhxiao*)

1.16.5 Other Improvements

- [0c737a94] - **release**: revert back master check (*hanhxiao*)
- [7b04697c] - **changelog**: revert the change log to empty (*hanhxiao*)
- [d02f320d] - revert docker file (*felix*)

CHAPTER 2

Tutorials

Warning: Tutorial is still under construction. Stay tuned! Meanwhile, we sincerely welcome you to contribute your own learning experience / case study with GNES!

2.1 Troubleshooting

2.1.1 Check if docker swarm/stack runs successfully

```
docker service ls
```

```
ID                  NAME                                MODE                REPLICAS
↪        IMAGE                                          PORTS
j7b533zxmzg5        gnes-swarm-2654_encoder             replicated          0/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:master
0vlxu4acg1ph        gnes-swarm-2654_income-proxy        replicated          0/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:master    *:4962->4962/tcp
equqrhsn7pky        gnes-swarm-2654_indexer             replicated          0/3
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:master
nd7euo7mcpa9        gnes-swarm-2654_middleman-proxy     replicated          0/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:master
ssdlk9gzmggw        gnes-swarm-2654_outgoing-proxy      replicated          0/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:master    *:4963->4963/tcp
xgxeetyhos6t        my-gnes_encoder                     replicated          1/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:a799a0f
zny37400p225        my-gnes_income-proxy                replicated          1/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:a799a0f   *:8598->8598/tcp
taqqg6qwrxlw        my-gnes_indexer                     replicated          3/3
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:a799a0f
j96gnny8ysbn        my-gnes_middleman-proxy             replicated          1/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:a799a0f
e28spnuksjw8        my-gnes_outgoing-proxy              replicated          1/1
↪        ccr.ccs.tencentyun.com/gnes/aipd-gnes:a799a0f   *:8599->8599/tcp
```

(continues on next page)

(continued from previous page)

In the above example, we started two swarms, i.e. `gnes-swarm-2654` and `my-gnes`. Unfortunately, `gnes-swarm-2654` fails to start and is not running at all. But how can one tell that?

Note the column `REPLICAS`, which indicates the number of running service (versus the number of required services). `gnes-swarm-2654` gives `0/0` for all services. This suggests the swarm fails to start. The next step is to investigate the reason.

2.1.2 Investigate the reason of a failed service

One can not print out all logs of a docker swarm. Instead, one can inspect service by service, e.g.

```
docker service ps gnes-swarm-2654_encoder --format "{{json .Error}}" --no-trunc
```

```
"\"invalid mount config for type \"bind\": bind source path does not exist: /data/han/
↪test-shell/output_data\""
"\"invalid mount config for type \"bind\": bind source path does not exist: /data/han/
↪test-shell/output_data\""
"\"invalid mount config for type \"bind\": bind source path does not exist: /data/han/
↪test-shell/output_data\""
"\"invalid mount config for type \"bind\": bind source path does not exist: /data/han/
↪test-shell/output_data\""
```

Now the reason is clear, `output_data` does not exist when starting the swarm. But why there are duplicated lines there? This is because docker swarm did three retries before giving up on starting this service, where each time it met the same problem. Thus four duplicated lines in total.

2.1.3 Delete a failed service

Now that the reason is clear, we can delete the failed service and release the resources.

```
docker stack rm gnes-swarm-2654
```

```
Removing service gnes-swarm-2654_encoder
Removing service gnes-swarm-2654_income-proxy
Removing service gnes-swarm-2654_indexer
Removing service gnes-swarm-2654_middleman-proxy
Removing service gnes-swarm-2654_outgoing-proxy
Removing network gnes-swarm-2654_gnes-net
```

2.1.4 Locate internal errors by looking at logs

Sometime the service fails to start but `docker service ps` gives no error,

```
docker service ps gnes-swarm-4254_encoder --format "{{json .Error}}" --no-trunc
```

```
""
```

Or it shows an error that is not explanatory.

```
"\"task: non-zero exit (2)\""
```

Often in this case, the service fails to start *not* due tothe docker config, but due to the GNES internal error. To see that,

```
docker service logs gnes-swarm-4254_income-proxy
```

```
gnes-swarm-4254_income-proxy.1.yj5v8n4dhfgv@VM-0-3-ubuntu    |                  [--
↪proxy_type {BS,Dict,MapProxyService,Message,MessageHandler,ProxyService,
↪ReduceProxyService,defaultdict}]
gnes-swarm-4254_income-proxy.1.yj5v8n4dhfgv@VM-0-3-ubuntu    |                  [--
↪batch_size BATCH_SIZE] [--num_part NUM_PART]
gnes-swarm-4254_income-proxy.1.kmgk21qo6m0n@VM-0-3-ubuntu    |                  [--
↪proxy_type {BS,Dict,MapProxyService,Message,MessageHandler,ProxyService,
↪ReduceProxyService,defaultdict}]
gnes-swarm-4254_income-proxy.1.w04d552cuj93@VM-0-3-ubuntu    | gnes proxy: error:␣
↪argument --batch_size: invalid int value: ''
gnes-swarm-4254_income-proxy.1.kmgk21qo6m0n@VM-0-3-ubuntu    |                  [--
↪batch_size BATCH_SIZE] [--num_part NUM_PART]
```

One can now clearly see that the error comes from an incorrectly given `--batch_size`, which throws from GNES CLI.

2.2 Protobuf Implementation

The file `gnes/proto/gnes.proto` defines the protobuf used in GNES. It is the core message protocol used in communicating between services. It also defines the interface of a gGRPC service.

`gnes_pb2.py` and `gnes_pb2_grpc.py` are python interfaces automatically generated by protobuf tools.

For developers who want to change the protobuf definition, one needs to first edit `gnes/proto/gnes.proto` and then regenerate the python codes (i.e. `gnes_pb2.py` and `gnes_pb2_grpc.py`).

2.2.1 Generating `gnes_pb2.py` and `gnes_pb2_grpc.py`

Take MacOS as an example,

1. Download `protoc-$VERSION-$PLATFORM.zip` from the official site and decompress it.
2. Copy the binary file and include to your system path:

```
cp ~/Downloads/protoc-3.7.1-osx-x86_64/bin/protoc /usr/local/bin/

cp -r ~/Downloads/protoc-3.7.1-osx-x86_64/include/* /usr/local/include/
```

1. Install gRPC tools dependencies: `brew install automake autoconf libtool`
2. Install gRPC and `grpc_python_plugin` from the source:

```
git clone https://github.com/grpc/grpc.git
git submodule update --init
make grpc_python_plugin
```

1. This will compile the grpc-python-plugin and build it to, e.g., `/Documents/grpc/bins/opt/grpc_python_plugin`
2. Generate the python codes:

```
SRC_DIR=gnes/proto/
PLUGIN_PATH=/Documents/grpc/bins/opt/grpc_python_plugin

protoc -I $SRC_DIR --python_out=$SRC_DIR --grpc_python_out=$SRC_DIR --plugin=protoc-
↪gen-grpc_python=${PLUGIN_PATH} ${SRC_DIR}gnes.proto
```

1. Fixing the import in `gnes_pb2_grpc.py`. For some reason (probably a bug of gRPC?), the generated code of `import` is not correct in `gnes_pb2_grpc.py`, you have to change it to the following:

```
# Generated by the gRPC Python protocol compiler plugin. DO NOT EDIT!
import grpc

from . import gnes_pb2 as gnes__pb2
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

P

Q

R

S

T

U

V

W

Y

Z