User manual for range tests

Written by
Zak Inglebright

Eberswalde, 17.09.2024

Initial requirements

In your chosen IDE, go to your terminal and type: `pip install -r dependencies.txt`.

Data_pipeline.run_pipeline() - version 0.2

class data_pipeline.run_pipeline(*username*, *password*, *sn_list*, *gtw_list*, *date_range*, *gtw_type*, *freq*, *to_file=None*)

The `data_pipeline.run_pipeline` class is designed to automate the extraction of data from Snowflake, process, summarize, and provide both summarized and unsummarized data outputs, alongside the visualisation of the aggregated data. This class is essential for handling large datasets for range tests, making it easier to analyze and interpret the performance of sensors to gateways.

Parameters:
username *(str, required)*: The Snowflake username needed for API authentication. This is a mandatory parameter to access the Snowflake database.
password *(str, required)*:

The Snowflake password corresponding to the username. This is also required for authentication purposes.

sn_list *(list of str, required)*:

A list of sensor identifiers. Each sensor ID should be a string and must end with a hyphen ('-') to avoid matching unwanted sensors. Example: ['n34-', 'n920-', 'n1093-'].

gtw_list *(list of str or int, default None)*:

For the SN2BG relation, a list of gateway identifiers, which represents border gateways (BG) must be provided. The gateway IDs can be specified either as strings (e.g., 'bg3-2401-28') or as integers when referring to their GTWEID. Note that gtw_list must consist entirely of either strings or integers, but not a mix of both. Additionally, for SN2MG relations, the user must enter integers, referring to their respective GTWEID.

date_range *(list of str, required)*:

The date range for the data extraction, specified as a list of two strings in the format 'YYYY-MM-DD'. This defines the start and end dates for the data retrieval.

gtw_type *(int, required)*:

This parameter defines the type of gateway relationship the pipeline will handle. It accepts only two values: 0 or 1. A value of 0 indicates an SN2BG (Sensor-to-Border Gateway) relationship. Conversely, a value of 1 indicates an SN2MG (Sensor-to-Mesh Gateway) relationship. This distinction is essential for the correct processing of the data based on the specific gateway type.

freq *(str, required)*:

This parameter specifies the temporal frequency for data aggregation. The user can choose from a range of standard frequency options, each represented by a single letter. For daily data aggregation, the user should input 'D', while 'W' will group the data on a weekly basis. Other standard time frequency inputs are accepted, but please refer here, for the full specification of available frequencies.

to_file *(str, default None):*

If specified, the function will export the resulting DataFrame (either SN2BG, SN2MG, their summaries, and plots) to the given file path. The file will be saved in CSV and PNG format. If this parameter is not provided, the data will not be saved to a file.

Returns:

(seaborn.barplot):

Once pipeline has been executed, several plots related to the RSSI, SNR and PER across different sensors to gateways overtime will be automatically produced.

SN2BG or SN2MG *(pandas.DataFrame)*:

Depending on the gtw_type specified, the function will return either the unsummarized raw data extracted from Snowflake for SN to BG relations (SN2BG) or for SN to MG relations (SN2MG).

SN2BG_summary or SN2MG_summary *(pandas.DataFrame)*:

Similarly, the summarized data, aggregated on a specified frequency, will either include metrics for SN to BG relations (SN2BG_summary) or for SN to MG relations (SN2MG_summary), based on the gtw_type provided. This summary includes metrics such as average RSSI, average SNR, total packets, and packet error rate.

Examples:

Creating a SN2BG_summary for sensors (4n9292, 4n9295, 4n9482) to BG (bg3-2401-28) on a daily frequency.

```
from WorkflowEngine import DataPipelineEngine
data_pipeline = DataPipelineEngine()

username='enter username'
password='enter password'
sn_list = ['4n9292-', '4n9295-', '4n9482-']
gtw_list = ['bg3-2401-28']
date_range = ['2024-09-06','2024-09-10']
gtw_type = 0
freq = 'D'


SN2BG, SN2BG_summary = data_pipeline.run_pipeline(username, password, sn_list,
gtw_list, date_range, gtw_type, freq)

SN2BG_summary.head()
```

sensor_id	bgtw_id	timestamp	avg_rssi	avg_snr	missing_pckts	total_pckts	pckt_error_rate
wf-4n9292	bg3-2401-28	2024-09-06	-65.00	13.80	14	17	82.35
wf-4n9292	bg3-2401-28	2024-09-07	-65.00	12.65	16	22	72.73
wf-4n9292	bg3-2401-28	2024-09-08	-65.00	12.55	12	23	52.17
.....	.....	.....	.....	.....	.....	.....	.....

Creating a SN2BG_summary for sensors (sn503, sn174, n727) to BG (31251, 21908) on a weekly frequency.

```
from WorkflowEngine import DataPipelineEngine
data_pipeline = DataPipelineEngine()

username='enter username'
password='enter password'
sn_list = ['n503-', 'n174-', 'n727-']
date_range = ['2024-08-06','2024-09-10']
gtw_list = [31251, 21908]
gtw_type = 0
freq = 'W'
```

```
SN2BG, SN2BG_summary= data_pipeline.run_pipeline(username, password, sn_list,
gtw_list, date_range,gtw_type,freq)

SN2BG_summary.head()
```

sensor_id	bgtw_id	timestamp	avg_rssi	avg_snr	missing_pckts	total_pckts	pckt_error_rate
sn-silvav3n174	bg-2m2n6	2024-08-11	-112.83	-0.19	18	36	50.00
sn-silvav3n174	bg-2m2n6	2024-09-08	-103.39	4.28	14	91	15.38
sn-silvav3n174	bg3-2401-6	2024-08-11	-113.18	3.50	15	53	28.30
.....	.....	.....	.....	.....	.....	.....	.....

Creating a SN2MG_summary dataframe for sensors (sn34, sn920, sn507, sn1049, sn130, sn1000) to MGs (30172, 2057) on a daily frequency.

```
username='enter username'
password='enter password'
sn_list = ['n34-', 'n920-', 'n1093-','n507-', 'n1049-','n130-', 'n1000-']
date_range = ['2024-09-06','2024-09-09']
gtw_list = [30172, 2057]
gtw_type = 1
freq = 'D'


SN2MG, SN2MG_summary= data_pipeline.run_pipeline(username, password, sn_list,
gtw_list, date_range,gtw_type,freq)

SN2MG_summary.head()
```

sensor_id	mgtw_id	timestamp	avg_rssi	avg_snr	SN2MG_distance _m	missing_pckts	total_pckts	pckt_error_rate
sn-silvav3n1049	mg2-9	2024-09-06	-80.00	9.50	84.55	0	1	0.00
sn-silvav3n1049	mg3-9	2024-09-06	-74.00	7.70	84.55	0	1	0.00
sn-silvav3n1093	mg2-9	2024-09-06	-87.75	8.68	58.26	7	11	63.64
.....	.....	.....	.....	.....	.....	.....	.....	.....

Saving data to the local hard drive.

```
username='enter username'
password='enter password'
sn_list = ['4n9292-', '4n9295-', '4n9482-']
gtw_list = ['bg3-2401-28']
date_range = ['2024-09-06','2024-09-10']
gtw_type = 0
freq = 'D'
path = 'your_directory/data'


SN2BG, SN2BG_summary = data_pipeline.run_pipeline(username, password, sn_list,
gtw_list, date_range, gtw_type, freq, to_file = path)
```

Notes:

Please refer here for source code.

Please refer here for more information on SN2MG relations.