Query API Tutorial

Query API Tutorial

This document explains how one could submit a CMap query via the API. We assume that you are familiar with the CLUE Query App, that you have read and internalized the following connectopedia articles ("How do I design a query?" (/connectopedia/how_to_construct_cmap_queries) and "Batch query tutorial" (connectopedia/batch_query_tutorial)), and that you know how to make RESTful API requests either through a client, like POSTMan or via the command line (like cURL).

We also assume that you are a registered user of CLUE and you have an API Key, see here (/api) for details.

EndPoint

The api endpoint is at https://api.clue.io/api/jobs (https://api.clue.io/api/jobs)

For L1000 queries

There are two ways of submitting a query on the server. If you have a file you would like to use as input for your gene lists, you can use a cURL command to upload it directly and run the query. If you have a list you would like to use, you can convert it into a stringified gmt and use a post request or cURL command to submit the input as a string in the request JSON.

Post request

Request payload looks like the following (part in bold means that they are required)

**Content-Type : **multipart/form-data

The following is the cURL command for submitting a query by uploading a file.

```
curl -i -X POST \ -H "user_key: XXXXXXXX" \ -H "Content-Type: multipart/form-data" \ -F 'tool_id=sig_gutc_tool' \ -F 'uptag-cmapfile=@/Users/foo/Downloads/uptag.gmt' \ -F "name=BAR" \ -F 'dntag-cmapfile=@/Users/foo/Downloads/dntag.gmt' \ -F 'ignoreWarnings': true \ -F "data_type=L1000" \ -F "dataset=Touchstone" api_url
```

Where api_url would be https://api.clue.io/api/jobs

**Content-Type : **application/json

```
"tool_id": "sig_gutc_tool",

"name": "(GSE32547) Pitavastatin treated HUVEC cells (1 uM at 4H) vs. DMSO treated",

"uptag-cmapfile":  "foo\t\bar\t\12345",

"data_type": "L1000",

"dataset": "Touchstone",
```

```
"ignoreWarnings": true,

"dntag-cmapfile":  "foo\t\bar\t\6789",

}```

**name**: the name of the query

**tool_id**: one of "sig_gutc_tool" or "sig_fastgutc_tool"

**uptag_cmapfile**: serialized gmt file of up-regulated genes or a local file

data_type: the data **type** of the **query** (Make **an** http GET request to [https://api.clue.io/api/dataTypes](https://api.clue.io/api/dataTypes) to get a list of available dataTypes. Default is "L1000"

dataset: the dataset to **query** against. **For** now only Touchstone is available (Make **an** http GET request to [https://api.clue.io/api/datasets](https://api.clue.io/api/datasets) to get a list of available datasets). Default is "Touchstone"

dntag_cmapfile: serialized gmt **file** of down-regulated genes or a **local file**

ignoreWarnings: whether to fail the request **if** the server produces warnings (**by** default it is **set** to true. I.**e** the request will be failed **if** there are warnings)

**Note: That the user_key is required in the header of the payload**

A cURL example of the above payload is below:

```curl -X POST --header "Content-Type: application/json" --header "Accept: application/json" --header "user_key: XXXXXX" -d "{
  \"tool_id\": \"sig_gutc_tool\",
  \"data_type\": \"L1000\",
  \"name\": \"(GSE32547) Pitavastatin treated HUVEC cells (1 uM at 4H) vs. DMSO treated\",
  \"uptag-cmapfile\":
\"TAG\t\t10365\t1831\t9314\t4846\t678\t22992\t3397\t26136\t79637\t5551\t7056\t79888\t1032\t51278\t64866\t29775\t994\t51696\t81839\t23580\t219654\t57178\t7014\t57513\t51599\t55818\t4005\t4130\t4851\t2050\t50650\t9469\t54438\t3628\t54922\t3691\t65981\t54820\t2261\t2591\t7133\t162427\t10912\t8581\t2523\t25807\t9922\t30850\t4862\t8567\t79686\t55615\t51283\t3337\t2887\t3223\t6915\t6907\t26056\t259217\t6574\t23097\t5164\t57493\t7071\t5450\t113146\t8650\",
  \"dntag-cmapfile\":
\"TAG\t\t5128\t5046\t956\t10426\t9188\t23403\t7204\t1827\t3491\t9076\t330\t8540\t22800\t10687\t19\t63875\t10979\t51154\t10370\t50628\t7128\t6617\t7187\t22916\t81034\t58516\t3096\t4794\t5202\t26511\t8767\t2355\t22943\t1490\t133\t11010\t51025\t23160\t56902\t3981\t5209\t6347\t5806\t7357\t9425\t3399\t6446\t64328\t6722\t8545\t688\t861\t390\t23034\t51330\t51474\t2633\t4609\",
   \"dataset\": \"Touchstone\"
```
```

}" "https://api.clue.io/api/jobs"```

Post response

If successful, HTTP code of 2XX will be sent to user with a link to where they can poll **for** the status/fetch the data **for display**

If failed, HTTP code of 4XX is sent with **an error** or warning message

Payload **for** a successful response looks like:

```{

"status": "pending",

"result": {

"job_id": "XXX",

"params": {

"tool_id": "sig_gutc_tool",

"data_type": "L1000",

"name": "test",

"dataset": "Touchstone",

"build_id": "a2geneid",

"tool_version": "1.1.1.2",

"Row_annot_path": "/cmap/data/Touchstone/gutc_background/annot/siginfo.txt",

"ignoreWarnings": false,

"numQueries": 1,

"undef_action": "warn",

"external_user_query": "true",

"es_tail": "up",

"uptag":
"http://data.clue.io/api/XXX@foo.org/results/Oct_04_2018/my_analysis.sig_gutc_tool.XXX/uptag.gmt",

"dntag":
"http://data.clue.io/api/XXX@foo.org/results/Oct_04_2018/my_analysis.sig_gutc_tool.XXX/dntag.gmt",
```

```
"config":
"http://data.clue.io/api/XXX@foo.org/results/Oct_04_2018/my_analysis.sig_gutc_tool.XXX/config.yaml"

},

"download_status": "completed",

"download_url":
"//s3.amazonaws.com/data.clue.io/api/XXX@foo.org/results/Oct_01_2018/my_analysis.sig_fastgutc_tool.XXX/my_analysis.sig_fastgutc_tool.XXX.tar.gz"

}

}```

**Error handling**

HTTP Status codes:

We **use** HTTP codes of 2XX to **mean** that the **query** was successfully submitted.

4XX **means** that there was a user **error in** the request

A code of 5XX **means** a system **error**

**Responses with 4XX status codes**

The structure of the **error** response object is **as** follows:

```{

"system": {

"warnings": [{"text" : "ABC"}],

"errors": [{"text" : "ABC"}]

},

"up": {

"warnings": [{"text" : "ABC"}],

"errors": [{"text" : "ABC"}]

},

"down": {

"warnings": [{"text" : "ABC"}],
```
```

"errors": [{"text" : "ABC"}]

},

"both": {

"warnings": [{"text" : "ABC"}],

"errors": [{"text" : "ABC"}]

}

}```

System errors/warnings are errors/warnings associated with the submission. For instance if a required field is absent you will get a system error.

Up **errors/warnings** are errors/warnings associated with the up geneset. For example you get an error if you submit a query that has less than 10 entrez gene IDs. You get a warning if you use gensets that are not in BING space

Down **errors/warnings** similar to "up" errors/warnings** **

Both error/warnings: If the error/warning is associated with both genesets. For instance, if there are overlaps in the up and down genesets, that is where the error will be reported.

To check **for** errors, always check the status code of the HTTP response. **If** it is 4XX then check the payload **for** the appropriate **error** messages.

Polling for completion

The response from a successful **query** should provide you with a job id (job_id). To **query for** the status of your job, **use** the endpoint at api/jobs/job_id, using the following curl command:

```curl -X GET --header "Accept: application/json" --header "user_key: XXX" "http://api.clue.io/api/jobs/findByJobId/job_idXXX"```

If the download_status is "completed," then you can download your results from the download_url using curl or a web browser.

```{

"status": "completed",

"job_id": "XXX",

"download_status": "completed",

"download_url":

```
"//s3.amazonaws.com/data.clue.io/api/XXX@foo.org/results/Oct_01_2018/my_analysis.sig_fastgutc_tool.XXX/my_analysis.sig_fastgutc_tool.XXX.tar.gz"

}```

Contents of downloaded folder

* cs_n1x476251.gct -- GCT version of the raw combined scores

* gutc_config.yaml -- The configuration file that was used to run the tool

* **matrices/**

* **gutc/**

* cs_sig.gctx - Connectivity scores

* ns_sig.gctx - Normalized score

* ns_pcl_cell.gctx - Normalized connectivity based on pcl and cell lines

* ns_pcl_summary.gctx - Summary of normalized scores for PCL

* ns_pert_cell.gctx

* ns_pert_summary.gctx

* ps_pcl_cell.gctx - percentile(tau) scores for pcl and cell lines

* ps_pcl_summary.gctx - percentile(tau) scores for pcl and cell lines

* ps_pert_cell.gctx

* ps_pert_summary.gctx

* query_info.txt

* **query**/

* cs_n1x476251.gctx - Raw combined connectivity scores

* cs_up_n1x476251.gctx - Raw up regulated connectivity scores

* cs_dn_n1x476251.gctx - Raw down regulated connectivity scores

* dn.gmt - The down genesets that you uploaded

* up.gmt - The up genesets that you uploaded

For more information **about** reading the gctx **format**, see [here]
(https://clue.io/cmapPy/pandasGEXpress.html).
```

For Proteomics queries

Like L1000 queries, there are **two** ways of submitting a proteomics **query** through the API: **file** upload and **post** request.

Uploading a file

The cURL command **for** uploading a **file for** a proteomics **query** looks like the following

```curl -i -X POST \

-H "user_key: xxxxxxxxxxx" \

-H "Content-Type: multipart/form-data"  \

-F "input_file-cmapfile=@/Users/foo/Downloads/GCP.gct" \

-F "assay=GCP" \

-F "name=BAR-P" \

-F "tool_id=sig_prot_query_tool" \

-F "introspect=true" \

-F "dataset=Touchstone-P" https://api.clue.io/api/jobs ```

**name**: the name of the query

**assay**: one of "P100" or "GCP

**input_file-cmapfile**: serialized gmt file or a local file

introspect: true or false value that determines **if** introspect will be calculated

fields_to_aggregate: array of fields to aggregate

**Post request**
Request payload looks like **one** of the following (part **in** bold **means** that they are required), including **an** example of a P100 payload and **an** example of a GCP payload

```{

"assay": "GCP",

"name": "GCP EZH2 LOF mutants (n=5)",

"input_file-cmapfile": "#1.3\n42\t5\t4\t10\nid\tpr_gcp_base_peptide\t...",

```
"dataset": "Touchstone-P",

"introspect"**: "**true**"**,

"tool_id": "sig_prot_query_tool",
}

{

"assay": "P100",

"name": "P100 MCF7 6H and 24H Jnk inhibitors (n=2 drugs)",

"Input_file-
cmapfile":"#1.3\n96\t12\t9\t26\nid\tpr_gene_id\tpr_gene_symbol\tpr_p100_base_pe
ptide\t...",

"tool_id": "sig_prot_query_tool",

"dataset": "Touchstone-P",

"fields_to_aggregate": ["cell_id"]
}```

Contents of downloaded folder
CONCATED_CONN.gct -- concatenated connectivity results

input_file.gct -- input file

INTROSPECT_CONN.gct -- concatenated introspect results

proteomics_config.yml -- proteomics configuration file

<div class='last_modified'>Last modified: Wed Jul 10 2019 16:38:43 GMT-0400
(EDT)</div>
<div style='display: none' data-docId='1Nyr8pOxRCtKxLxO81_0KnMjovlqN-
aTqpwf39TBtbR8' class='docId'></div>
```