SigRepo Function Suite

Monti Lab

2025-02-05

Introduction

The `SigRepo` package includes a suite of functions for easily storing and managing biological signatures and its constituents. Currently, `Sigrepo` is capable of storing, searching, and retrieving signatures and its signature collections from a MySQL Database of choice. See here for documentation of how set-up the MySQL database with the appropriate schema.

In order to interact with a suite of functions in `SigRepo` package, the input data must be in a format of R6 objects for the representation of signatures and signature collections, and they can be created using our proprietary package, OmicSignature.

For more information click the links below.

- Overview of the object structure
- Create an OmicSignature (OmS)
- Create an OmicSignatureCollection (OmSC)

For demonstrations, we will walk through the steps of how to use `SigRepo` package to store, retrieve, and interact with a list of signatures stored in our MySQL SigRepo Database.

Installation

- Using `devtools` package

```
# Load devtools package
library(devtools)

# Install SigRepo
devtools::install_github(repo = 'montilab/SigRepo')

# Install OmicSignature
devtools::install_github(repo = 'montilab/OmicSignature')
```

Connect to SigRepo Database

We adopted a MySQL Database structure for efficiently storing, searching, and retrieving the biological signatures and its constituents. To access the signatures stored in our database, you MUST register here to create an account or contact our admin to be added.

There are three types of user accounts: - `admin` has READ and WRITE access to all signatures in the database. - `editor` has READ and WRITE access to ONLY their own uploaded signatures in the database. - `viewer` has ONLY READ access to view a list of signatures in the database but DO NOT HAVE WRITE access to the database.

Once you have a valid account, to connection to our SigRepo Database, one can use `newConnHandler()` function to create a handler which will contain appropriate credentials to establish connection to our database.

```
# Create a connection handler
conn_handler <- SigRepo::newConnHandler(
  dbname = Sys.getenv("DBNAME"),
  host = Sys.getenv("HOST"),
  port = as.integer(Sys.getenv("PORT")),
  user = Sys.getenv("USER"),
  password = Sys.getenv("PASSWORD")
)
```

Load Signatures

Here, we provided two signature objects that came with the package for demonstrations:

1. omic_signature_1
2. omic_signature_2

```
# Getting the signature path
signature_path <- base::system.file("inst/data/signatures", package = "SigRepo")

# Read in the signature object
omic_signature_1 <- base::readRDS(file.path(signature_path, "omic_signature_1.RDS"))
omic_signature_2 <- base::readRDS(file.path(signature_path, "omic_signature_2.RDS"))
```

Create a signature collection

Here, we will create a signature collection object using **OmicSignature** package

```
# Create collection metadata
metadata <- list(
  "collection_name" = "Example_Collection",
  "description" = "An example of signature collection",
  "author" = "me"
)

omic_collection <- OmicSignature::OmicSignatureCollection$new(
  OmicSigList = list(omic_signature_1, omic_signature_2),
  metadata = metadata
)
#>   [Success] OmicSignature Collection Example_Collection created.
```

Upload a collection

The function `addSignatureCollection()` allows the user to upload a collection to the database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function.

```
# Add collection to database
SigRepo::addSignatureCollection(
  conn_handler = conn_handler,
  omic_collection = omic_collection
)
#> Uploading each signature in the collection into the database...
#>  You already uploaded a signature with signature_name = 'Myc_reduce_mice_liver_24m_v1' into the SigR
#>  Use searchSignature() to see more details about the signature.
#>  To re-upload, try to use a different name.
#>  ID of the uploaded signature:
#>  You already uploaded a signature with signature_name = 'Myc_reduce_mice_liver_24m_v2' into the SigR
#>  Use searchSignature() to see more details about the signature.
#>  To re-upload, try to use a different name.
#>  ID of the uploaded signature:
#> Uploading collection metadata into the database...
#> Adding user to collection access table in the database...
#> Adding signature to collection access table in the database...
#> Finished uploading.
#> [1] TRUE
```

Search for a collection

The function `searchCollection()` allows the user to search for all or a specific signature that is available in the database.

- Search for all signatures

```
knitr::kable(
  SigRepo::searchCollection(
    conn_handler = conn_handler
  ),
  row.names = FALSE
)
```

collection_id	collection_name	description	user_name	date_created	collection_hashkey	signature_id	signature_collection_hashkey	signature_name
17	Example_Collection	An example of signature collection	root	2025-02-05 11:14:08	[illegible]	[illegible]	[illegible]	Myc_reduce_mice_liver_24
17	Example_Collection	An example of signature collection	root	2025-02-05 11:14:08	[illegible]	[illegible]	[illegible]	Myc_reduce_mice_liver_24

- Search for a specific collection, e.g., **"Example_Collection"**, in the database

```
knitr::kable(
  SigRepo::searchCollection(
    conn_handler = conn_handler,
    collection_name = "Example_Collection"
  ),
  row.names = FALSE
)
```

collection_id	collection_name	description	user_name	date_created	collection_hash	signature_id	signature_collection	signature_hash	key_name
17	Example_Collection	Collection of signature collection	root	2025-02-05 11:14:08	a6b91132ecc6b10e5d26b2	26	17	7d1	liver_24
17	Example_Collection	Collection of signature collection	root	2025-02-05 11:14:08	a6b91132ecc6b10e5d26b2	27	17	6e8	liver_24

Update a collection

The function `updateCollection()` allows the user to update a collection in the sigrepo database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function. Furthermore, the user can **ONLY UPDATE** their own uploaded signatures or was given the `editor` permission from other users to do so.

For example, if you wish to replace the description of **"Example_Collection"** in the database

```
# Search for Example_Collection in the database
collection_tbl <- SigRepo::searchCollection(
  conn_handler = conn_handler,
  collection_name = "Example_Collection"
)

# Get the collection id
collection_id <- unique(collection_tbl$collection_id)

# Updating the collection with a new description
SigRepo::updateCollectionMetadata(
  conn_handler = conn_handler,
  collection_id = collection_id,
  description =  "This is the updated description."
)
#> collection_id = '17' has been updated.
#> [1] TRUE
```

```
knitr::kable(
  SigRepo::searchCollection(
    conn_handler = conn_handler,
    collection_name = "Example_Collection"
  ),
  row.names = FALSE
)
```

collection_id	collection_name	description	user_name	date_created	collection_hashkey	signature_id	signature_collection_id	signature_hashkey
17	Example_Collection	This is the updated description.	root	2025-02-05 11:14:08	a6b91132ecc6b10e5d06b2	2	17	liver_24
17	Example_Collection	This is the updated description.	root	2025-02-05 11:14:08	a6b91132ecc6b10e5d06b2	3	17	liver_24

Delete a collection

The function `deleteCollection()` allows the user to delete a collection from the database.

IMPORTANT NOTE: The user must have `editor` or `admin` access to use this function. Furthermore, the user can **ONLY DELETE** their own uploaded collections or was given the `editor` permission from other users to do so.

```
# Search for Example_Collection in the database
collection_tbl <- SigRepo::searchCollection(
  conn_handler = conn_handler,
  collection_name = "Example_Collection"
)

# Get the collection id
collection_id <- unique(collection_tbl$collection_id)

# Remove collection from the database
SigRepo::deleteCollection(
  conn_handler = conn_handler,
  collection_id = collection_id
)
#> Remove collection_id = '17' from 'collection' table of the database.
#> Remove collection_id = '17' from 'collection_access' table of the database.
#> Remove signatures belongs to collection_id = '17' from 'signature_collection_access' table of the da
#> collection_id = '17' has been removed.
#> [1] TRUE
```