Co-funded by the Horizon 2020 Framework Programme of the European Union

Grant agreement No. 671555

ExCAPE
Exascale Compound Activity Prediction Engines

Future and Emerging Technologies (FET)

Call: H2020-FETHPC-2014
Topic: FETHPC-1-2014
Type of action: RIA

Deliverable D3.14

Workflows report
Overview of developed methods and workflows

Due date of deliverable: 31.12.2017 (M27)
Actual submission date: 21.09.2018

Start date of Project: 1.9.2015 Duration: 36 months

Responsible Consortium Partner: IDEA
Contributing Consortium Partners: JP, AZ
Name of author(s): Nina Jeliazkova (IDEA), Hongming Chen (AZ), Noe Sturm (AZ), Vladimir Chupakhin (JP), Vedrin Jeliazkov (IDEA)
Internal Reviewer(s): Tom Vander Aa (IMEC)
Revision: V1.3

Project co-funded by the European Union within the H2020 Framework Programme (2014-2020)		
Dissemination Level		
PU	Public	**PU**
PP	Restricted to other programme participants (including the Commission Services	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Document revision tracking

This page is used to follow the deliverable production from its first version until it has been reviewed by the assessment team. Please give details in the table below about successive releases.

Release number	Date	Reason of this release and/or validation	Dissemination of this release (task level, WP/ST level, Project Office Manager, Industrial Steering Committee, etc)
V1.0	2.03.2018	First draft for discussion	WP3 project partner
V1.1	24.07.2018		WP3 project partner
V1.2	28.08.2018	Partner contributions	WP3 project partner
V1.3	18.09.2018	Final version	PU

Glossary

NCM	Non-Conformity Measure
XGBoost	parallel tree boosting library
SVM	Support Vector Machine
RF	Random Forest
SMURFF	Scalable Matrix Factorization Framework

Link to Tasks

Task number	Work from task carried out	Deviations from task technical content, with motivation and summary of impact
T3.6	Development of prediction workflows	The development of prediction workflows was delayed.

Table of Contents

1 Executive summary

This report provides an overview of the workflows used to benchmark multiple algorithms enabling chemogenomics predictions on ExCAPE-DBv5 datasets. Multitask (SMURFF, ExNET) and single task (xgboost, SVM, Random Forest) machine learning methods were used.

Although the bulk of the work for this deliverable occurred earlier on in the project, the final write-up of all of the workflows was delayed until considerably later than expected because there is workflows were updated until quite late in the project. However, the delaying of the write-up resulted in little practical impact given that the workflows were available and being used.

2 Introduction – Aim

Methods behind developed algorithms were described in WP1 and WP2. This reports provides an overview and links to the relevant deliverables and describes the results of packaging of the machine learning algorithms developed by into reproducible workflows using Hyperloom package [1], command line interface and Jupyter interactive compute notebooks [2].

3 Methods

3.1 Matrix factorization

ExCAPE matrix factorization (MF) framework SMURFF was previously described in deliverables D2.4, D2.7, D2.12. The open source code is available on https://github.com/ExaScience/smurff. Comparison with other MF methods is reported in D2.18.

Developed code can be compiled specifically to the computer architecture or installed using conda, a package, dependency and environment manager. WP2 team developed a *conda* [3] build recipe, thus only one command required to install a software and all required dependencies: *conda install -c vanderaa smurff*. This significantly eases the deployment of the SMURFF on both Linux and Windows.

3.2 Deep learning

ExNET is an implementation of a neural network architecture (feed forward fully connected layers) delivered as prototypes by WP1 in deliverables D1.4 and D1.5. Originally, the network was implemented by JKU binet [4], while ExNET is reimplemented using the Google Tensorflow framework in a python script as reported in D2.11. The ExNET python script allows to build, train, test deep neural net models with customizable scoring functions and to make predictions with the trained models. The code is available through ExCAPE's private git code repository (*git:code/wp2/exnet*). The implementation of ExNET was first developed and later extended (April 2018) to support dense descriptors such as the new chem2vec molecular descriptors (introduced in D1.6). Full-scale validation and benchmarking will be reported by D3.18.

3.3 Hyperloom: a platform for defining and executing scientific pipelines in distributed environments

Every ML algorithm used in the project ExCAPE (except SMURFF) was run on IT4I using the Hyperloom package developed in D2.8. Hyperloom is used to distribute multiple runs in nested-cross validation routines. This includes a hyperparameter search with one training and one testing run per hyperparameter and a further training and testing for selected models using best hyperparameters.

The ExNET hyperloom pipeline is available at ExCAPE git (*git:code/wp2/loom-pipelines/loom-exnet*)

3.4 Single task learners

To train single task machine learning models, the pipeline had one further level of parallelization, since hyperparameters were searched individually for each target, hence resulting in evaluation models specific to each target. The used pipeline for single-task learners is described in D2.11. It relies on Hyperloom [1] and allows to easily integrate different single-task algorithms, e.g. XGBoost and SVM. Hyperloom automatically splits each submitted task into several “mini” tasks which are the different components of a computational graph. This graph sorts out when each “mini” task can be run relative to the overall task by considering which task can be computed in parallel. Therefore, it optimally uses computing resources of IT4I HPC center. It is particularly useful in performing multiple parallel tasks. One other strength is that it is designed to be user-friendly, hence one do not require to explicitly specify all the details of the computational graph. Available at ExCAPE git *code\wp2\loom-pipelines\loom-xgboost*

3.5 Conformal prediction

Conformal Predictors (D1.4) take Non-Conformity Measures (which express how non-conforming a given example appears compared to those in the training set) and, given a significance level, produce Region Predictions, i.e. subsets of the label space (rather a single label). The conformal predictors could take NCM calculated by arbitrary ML method. The NCMs are obtained for a calibration set. Finally, NCMs are calculated for each test object and for each possible value that the label of test object might take. D1.4. recommends Mondrian Inductive Conformal predictors for ExCAPE. Reference implementation as Jupyter notebook is available at ExCAPE *git:excape/code/wp1/MICPv2.*

4 Workflows

The HPC workflows in ExCAPE are defined and run via HyperLoom – an open source, HPC solution for defining and executing scientific workflows [1]. The Hyperloom Python interface for defining tasks is used to build and execute machine learning pipelines for the methods described in the previous section on ExCAPE-DBv5 with nested cross-validation for hyperparameter search (D2.20).
To bring the HPC experience closer to less experienced users, a REST API and web interface for running and monitoring complex HPC jobs was developed (D2.16) It enables interaction with the HPC through a web browser or scientific notebooks.

4.1 Data preparation

ExCAPE-ML is a subset of ExCAPE-DB designed especially for large scale machine learning experiments. The dataset was prepared as a set of three files (activities.txt, clusters.txt, folds.txt) once and used to prepare the input files for each ML algorithms. The overall preparation script is available on git:

excape\code\wp3\data_preparation_v5

4.1.1 The 300-75-75 rule

We only included targets for modeling that satisfy all of the following criteria: at least 300 compounds, at least 75 active compounds and at least 75 inactive compounds for 10uM activity cutoff (level 5), all other targets were not included in the modeling exercises. The resulting file is available as a text file containing three columns: compound identifier, target name, pXC50 activity value.

The file is available on Salomon at:
/scratch/work/project/excape-public/data_release_v5/version5/activities.txt.gz

4.1.2 Three folds split

There dataset resulting from 4.1.1 was split into three separate folds based on a structural clustering of the compounds, that is, each chemical cluster is present in only one fold, either train or test or validation. This concept of cluster-based-cross-validation allows better generalization of the models to be more sensitive for the new chemical compounds. We distributed clusters in each fold by making sure relatively even folds sizes and all the targets present in each fold. There are two resulting files. The first file contains each compounds assignment to a cluster with two columns: compound identifier and cluster identifier. The second file contains the assignment of each cluster to one of the three folds with two columns: cluster identifier and fold identifier.

Clusters:
/scratch/work/project/excape-public/data_release_v5/version5/clustering.txt.gz

Folds:
/scratch/work/project/excape-public/data_release_v5/version5/folds.txt.gz

4.1.3 Nested -cross validation

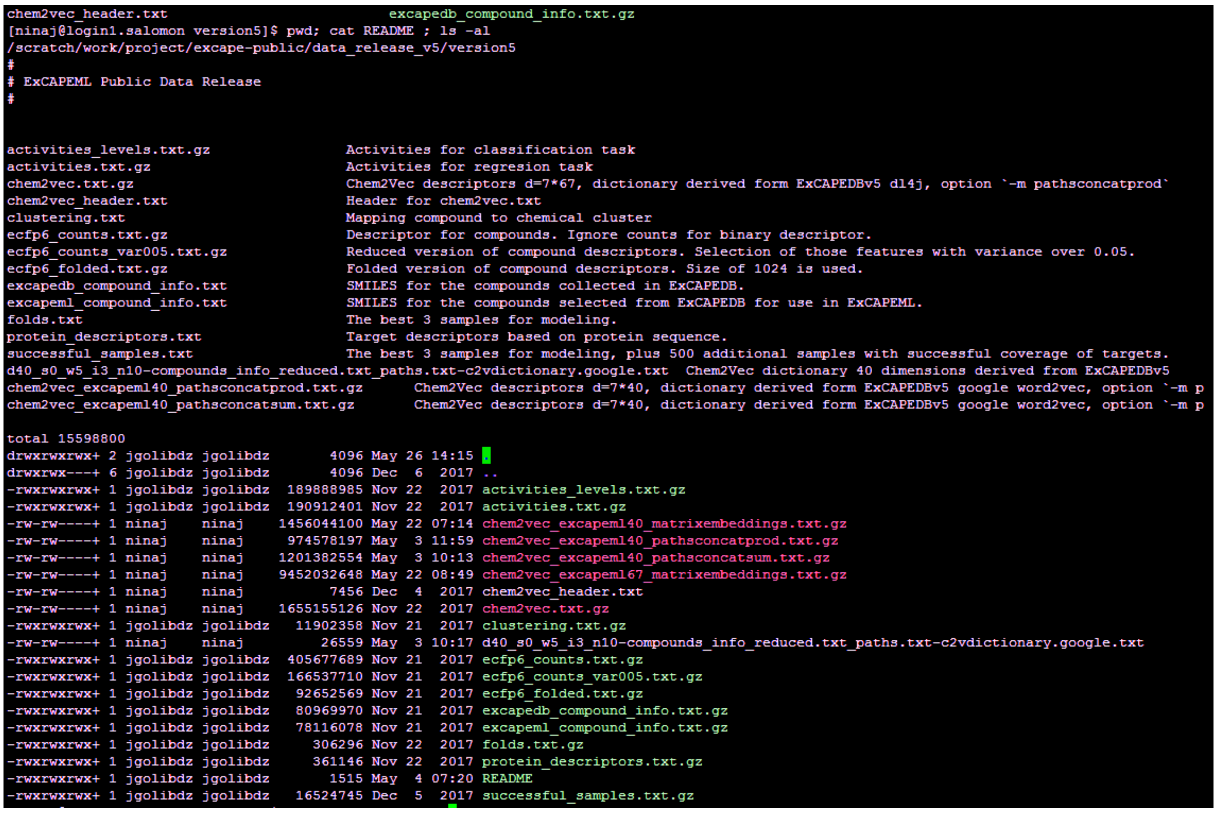
The three folds were used in a three-fold nested cross validation routine to evaluate and select the best hyperparameters; such large splits on the train, test and validation were used for two reasons: realistic and harsh modeling environment that will evaluate performance of the multi-task algorithms over single task machine learning algorithms. Another one is limitation on the number of cross-validation runs due to the required computer time, as increasing the number of the folds will increase number of required compute cycles.

The data release is at:

/scratch/work/project/excape-public/data_release_v5/version5 (**Figure 1**, Table 1).

activities_levels.txt.gz	Activities for classification task
activities.txt.gz	Activities for regression task
clustering.txt	Mapping compound to chemical cluster
ecfp6_counts.txt.gz	Descriptor for compounds. Ignore counts for binary descriptor.
ecfp6_counts_var005.txt.gz	Reduced version of compound descriptors. Selection of those features with variance over 0.05.
ecfp6_folded.txt.gz	Folded version of compound descriptors. Size of 1024 is used.
excapedb_compound_info.txt	SMILES for the compounds collected in ExCAPEDB
excapeml_compound_info.txt	SMILES for the compounds selected from ExCAPEDB for use in ExCAPEML
folds.txt	The best 3 samples for modeling
protein_descriptors.txt	Target descriptors based on protein sequence
successful_samples.txt	The best 3 samples for modeling, plus 500 additional samples with successful coverage of targets
chem2vec.txt.gz	Chem2Vec descriptors d=7*67, dictionary derived from ExCAPEDBv5 dl4j, option `-m pathsconcatprod`
chem2vec_header.txt	Header for chem2vec.txt
chem2vec_excapeml40_pathsconcatprod.txt.gz	Chem2Vec descriptors d=7*40, dictionary derived from ExCAPEDBv5 google word2vec, option `-m pathsconcatprod`
chem2vec_excapeml40_pathsconcatsum.txt.gz	Chem2Vec descriptors d=7*40, dictionary derived from ExCAPEDBv5 google word2vec, option `-m pathsconcatsum`

Table 1. ExCAPE data release v5 files description.



```
chem2vec_header.txt                                 excapedb_compound_info.txt.gz
[ninaj@login1.salomon version5]$ pwd; cat README ; ls -al
/scratch/work/project/excape-public/data_release_v5/version5
#
# ExCAPEML Public Data Release
#


activities_levels.txt.gz                Activities for classification task
activities.txt.gz                       Activities for regresion task
chem2vec.txt.gz                         Chem2Vec descriptors d=7*67, dictionary derived form ExCAPEDBv5 dl4j, option `-m pathsconcatprod`
chem2vec_header.txt                     Header for chem2vec.txt
clustering.txt                          Mapping compound to chemical cluster
ecfp6_counts.txt.gz                     Descriptor for compounds. Ignore counts for binary descriptor.
ecfp6_counts_var005.txt.gz              Reduced version of compound descriptors. Selection of those features with variance over 0.05.
ecfp6_folded.txt.gz                     Folded version of compound descriptors. Size of 1024 is used.
excapedb_compound_info.txt              SMILES for the compounds collected in ExCAPEDB.
excapeml_compound_info.txt              SMILES for the compounds selected from ExCAPEDB for use in ExCAPEML.
folds.txt                               The best 3 samples for modeling.
protein_descriptors.txt                 Target descriptors based on protein sequence.
successful_samples.txt                  The best 3 samples for modeling, plus 500 additional samples with successful coverage of targets.
d40_s0_w5_i3_n10-compounds_info_reduced.txt_paths.txt-c2vdictionary.google.txt  Chem2Vec dictionary 40 dimensions derived from ExCAPEDBv5
chem2vec_excapeml40_pathsconcatprod.txt.gz      Chem2Vec descriptors d=7*40, dictionary derived form ExCAPEDBv5 google word2vec, option `-m p
chem2vec_excapeml40_pathsconcatsum.txt.gz       Chem2Vec descriptors d=7*40, dictionary derived form ExCAPEDBv5 google word2vec, option `-m p

total 15598800
drwxrwxrwx+ 2 jgolibdz jgolibdz       4096 May 26 14:15 .
drwxrwx---+ 6 jgolibdz jgolibdz       4096 Dec  6  2017 ..
-rwxrwxrwx+ 1 jgolibdz jgolibdz  189888985 Nov 22  2017 activities_levels.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz  190912401 Nov 22  2017 activities.txt.gz
-rw-rw----+ 1 ninaj    ninaj    1456044100 May 22 07:14 chem2vec_excapeml40_matrixembeddings.txt.gz
-rw-rw----+ 1 ninaj    ninaj     974578197 May  3 11:59 chem2vec_excapeml40_pathsconcatprod.txt.gz
-rw-rw----+ 1 ninaj    ninaj    1201382554 May  3 10:13 chem2vec_excapeml40_pathsconcatsum.txt.gz
-rw-rw----+ 1 ninaj    ninaj    9452032648 May 22 08:49 chem2vec_excapeml67_matrixembeddings.txt.gz
-rw-rw----+ 1 ninaj    ninaj          7456 Dec  4  2017 chem2vec_header.txt
-rw-rw----+ 1 ninaj    ninaj    1655155126 Nov 22  2017 chem2vec.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz   11902358 Nov 21  2017 clustering.txt.gz
-rw-rw----+ 1 ninaj    ninaj         26559 May  3 10:17 d40_s0_w5_i3_n10-compounds_info_reduced.txt_paths.txt-c2vdictionary.google.txt
-rwxrwxrwx+ 1 jgolibdz jgolibdz  405677689 Nov 21  2017 ecfp6_counts.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz  166537710 Nov 21  2017 ecfp6_counts_var005.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz   92652569 Nov 21  2017 ecfp6_folded.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz   80969970 Nov 21  2017 excapedb_compound_info.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz   78116078 Nov 21  2017 excapeml_compound_info.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz     306296 Nov 22  2017 folds.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz     361146 Nov 22  2017 protein_descriptors.txt.gz
-rwxrwxrwx+ 1 jgolibdz jgolibdz       1515 May  4 07:20 README
-rwxrwxrwx+ 1 jgolibdz jgolibdz   16524745 Dec  5  2017 successful_samples.txt.gz
```

Figure 1. Data release folder at Salomon *scratch/work/project/excape-public/data_release_v5/version5/*

4.1.4 Chemical structure standardization

The chemical structure standardization is performed by open source ambitcli software, previously described in ExCAPE-DB publication [6]. The latest versions are downloadable from http://ambit.sourceforge.net/ambitcli_standardisation.html . While shell scripts and trivial parallelization were used in the beginning of the project for this task, it is also possible to use Hyperloom and/or Jupyter notebooks (demo notebook for standardization at https://github.com/ideaconsult/apps-ambit/blob/master/standardize/ambit_standardize_demo.ipynb)

4.1.5 Descriptors

The ECFP fingerprints are also calculated using *ambitcli*, using the CDK Circular Fingerprinter class (see the relevant options at http://ambit.sourceforge.net/download_ambitcli.html). A functionality to extract substructure pattern corresponding to an ECFP fingerprint was contributed to the CDK library and included in CDK 1.5.14 (released Oct 2016, https://github.com/cdk/cdk/wiki/1.5.14-Release-Notes)

The code to calculate chem2vec descriptors is at git:*excape/code/wp3/chem2vec* , which includes shell scripts calling the compiled ambit-search jar (code at git:*excape/code/wp3/chem2vec/ambit-search*). The *ambit-search* code also contains memory efficient implementation of sphere clustering, which was used to prepare the crossvalidation folds. The ambit-search code will be released under open source license upon publication.

4.1.6 Matrix market format

Input files were prepared and formatted as SciPy [5] sparse matrices using the matrix market format. This format was selected because it fits well chemogenomics data and is a relatively efficient was to store and load highly sparse data such as: ECFP descriptors for a set of compounds, and bioactivity data. There are two python scripts to prepare the input files. The first one loads the activites.txt.gz, folds.txt.gz and clustering.txt.gz files and prepare folds matrix market files and sets up a folder and file architecture facilitating the nested-cross validation routine (figure 2 and figure 3). The second script generates a matrix market file from the molecular descriptors as provided in IJV format (compound identifier, hash key, value).

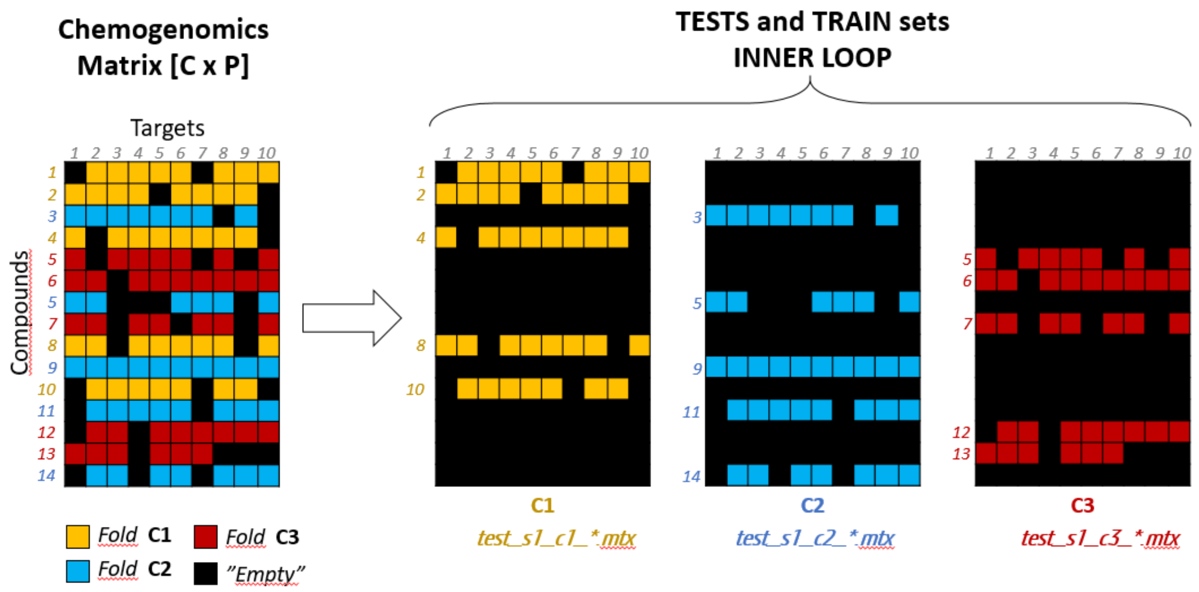


Figure 2. Illustration of the three bioactivity folds used as train/test sets inner loop of the nested cross validation. These folds are also used as test sets in the outer loop of the nested cross validations.

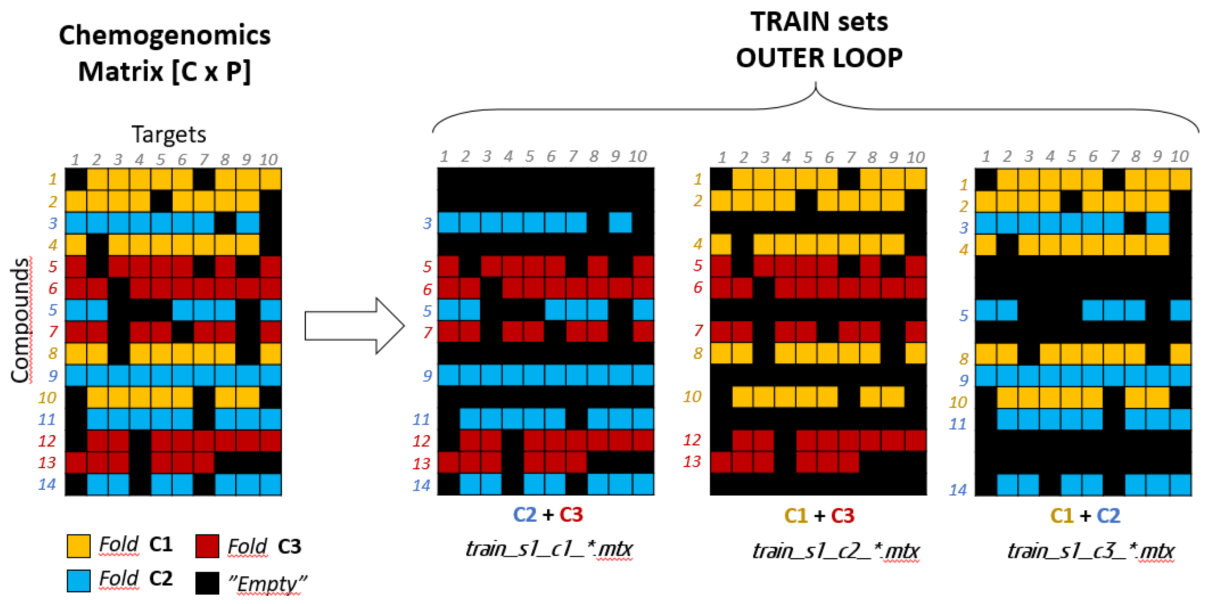
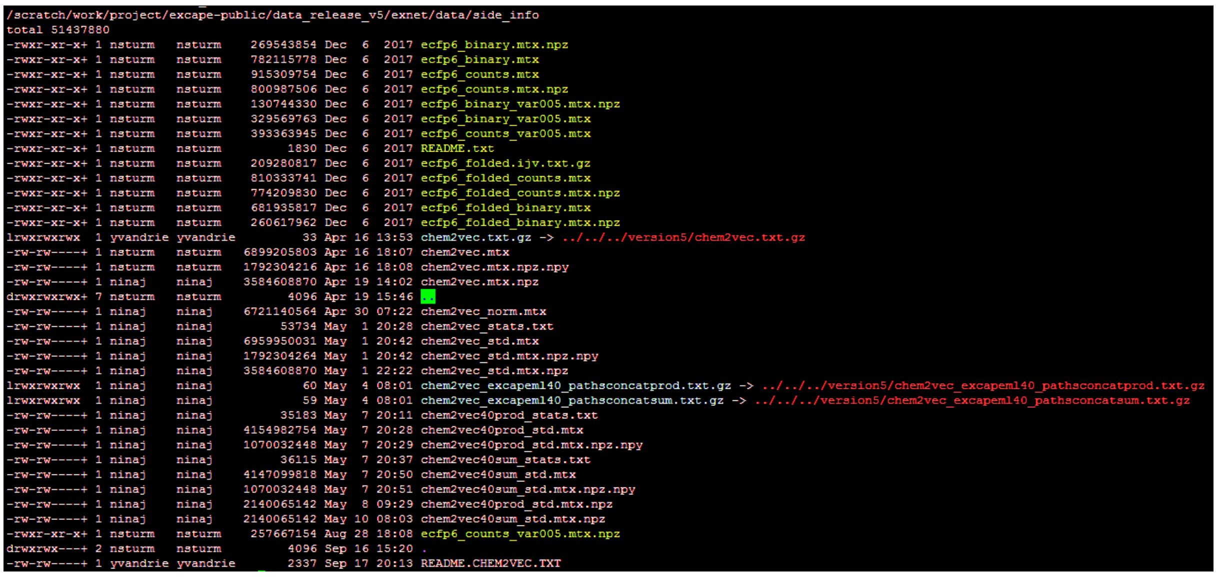


Figure 3. Illustration of the training sets used in the outer loop of the nested cross validation.

Code for data preparation in format suitable for machine learning workflows (e.g. Matrix Market Format) is available at *git: excape\code\wp3\data_preparation_v5* . The folder at Salomon containing chemical structure descriptors (*/scratch/work/project/excape-public/data_release_v5/exnet/data/side_info*) is shown on Figure 4.



```
/scratch/work/project/excape-public/data_release_v5/exnet/data/side_info
total 51437880
-rwxr-xr-x+ 1 nsturm   nsturm    269543854 Dec  6  2017 ecfp6_binary.mtx.npz
-rwxr-xr-x+ 1 nsturm   nsturm    782115778 Dec  6  2017 ecfp6_binary.mtx
-rwxr-xr-x+ 1 nsturm   nsturm    915309754 Dec  6  2017 ecfp6_counts.mtx
-rwxr-xr-x+ 1 nsturm   nsturm    800987506 Dec  6  2017 ecfp6_counts.mtx.npz
-rwxr-xr-x+ 1 nsturm   nsturm    130744330 Dec  6  2017 ecfp6_binary_var005.mtx.npz
-rwxr-xr-x+ 1 nsturm   nsturm    329569763 Dec  6  2017 ecfp6_binary_var005.mtx
-rwxr-xr-x+ 1 nsturm   nsturm    393363945 Dec  6  2017 ecfp6_counts_var005.mtx
-rwxr-xr-x+ 1 nsturm   nsturm         1830 Dec  6  2017 README.txt
-rwxr-xr-x+ 1 nsturm   nsturm    209280817 Dec  6  2017 ecfp6_folded.ijv.txt.gz
-rwxr-xr-x+ 1 nsturm   nsturm    810333741 Dec  6  2017 ecfp6_folded_counts.mtx
-rwxr-xr-x+ 1 nsturm   nsturm    774209830 Dec  6  2017 ecfp6_folded_counts.mtx.npz
-rwxr-xr-x+ 1 nsturm   nsturm    681935817 Dec  6  2017 ecfp6_folded_binary.mtx
-rwxr-xr-x+ 1 nsturm   nsturm    260617962 Dec  6  2017 ecfp6_folded_binary.mtx.npz
lrwxrwxrwx  1 yvandrie yvandrie         33 Apr 16 13:53 chem2vec.txt.gz -> ../../../version5/chem2vec.txt.gz
-rw-rw----+ 1 nsturm   nsturm   6899205803 Apr 16 18:07 chem2vec.mtx
-rw-rw----+ 1 nsturm   nsturm   1792304216 Apr 16 18:08 chem2vec.mtx.npz.npy
-rw-rw----+ 1 ninaj    ninaj    3584608870 Apr 19 14:02 chem2vec.mtx.npz
drwxrwxrwx+ 7 nsturm   nsturm         4096 Apr 19 15:46 ..
-rw-rw----+ 1 ninaj    ninaj    6721140564 Apr 30 07:22 chem2vec_norm.mtx
-rw-rw----+ 1 ninaj    ninaj         53734 May  1 20:28 chem2vec_stats.txt
-rw-rw----+ 1 ninaj    ninaj    6959950031 May  1 20:42 chem2vec_std.mtx
-rw-rw----+ 1 ninaj    ninaj    1792304264 May  1 20:42 chem2vec_std.mtx.npz.npy
-rw-rw----+ 1 ninaj    ninaj    3584608870 May  1 22:22 chem2vec_std.mtx.npz
lrwxrwxrwx  1 ninaj    ninaj            60 May  4 08:01 chem2vec_excapeml40_pathsconcatprod.txt.gz -> ../../../version5/chem2vec_excapeml40_pathsconcatprod.txt.gz
lrwxrwxrwx  1 ninaj    ninaj            59 May  4 08:01 chem2vec_excapeml40_pathsconcatsum.txt.gz -> ../../../version5/chem2vec_excapeml40_pathsconcatsum.txt.gz
-rw-rw----+ 1 ninaj    ninaj         35183 May  7 20:11 chem2vec40prod_stats.txt
-rw-rw----+ 1 ninaj    ninaj    4154982754 May  7 20:28 chem2vec40prod_std.mtx
-rw-rw----+ 1 ninaj    ninaj    1070032448 May  7 20:29 chem2vec40prod_std.mtx.npz.npy
-rw-rw----+ 1 ninaj    ninaj         36115 May  7 20:37 chem2vec40sum_stats.txt
-rw-rw----+ 1 ninaj    ninaj    4147099818 May  7 20:50 chem2vec40sum_std.mtx
-rw-rw----+ 1 ninaj    ninaj    1070032448 May  7 20:51 chem2vec40sum_std.mtx.npz.npy
-rw-rw----+ 1 ninaj    ninaj    2140065142 May  8 09:29 chem2vec40prod_std.mtx.npz
-rw-rw----+ 1 ninaj    ninaj    2140065142 May 10 08:03 chem2vec40sum_std.mtx.npz
-rwxr-xr-x+ 1 nsturm   nsturm    257667154 Aug 28 18:08 ecfp6_counts_var005.mtx.npz
drwxrwx---+ 2 nsturm   nsturm         4096 Sep 16 15:20 .
-rw-rw----+ 1 yvandrie yvandrie       2337 Sep 17 20:13 README.CHEM2VEC.TXT
```

Figure 4. Data release folder with chemical descriptors in machine learning friendly formats

4.1.7 Hyperloom pipelines

All the used hyperloom pipelines are stored, with one folder per machine learning algorithm and similar architectures per pipeline.

The folders can be found at: /scratch/work/project/excape-public/pipelines

Each pipeline consists mainly of, a python pipeline script invoking *hyperloom* and creating the training, test and prediction tasks; a PBS script to submit the jobs to the scheduling system and possible configuration files containing parameters or options depending on the machine learning algorithm used.

Exnet_v3 pipeline (provided by WP2):
/scratch/work/project/excape-public/pipelines/edlp_ecfp/

XGboost: /scratch/work/project/excape-public/pipelines/xbg_ecfp/

SVM: /scratch/work/project/excape-public/pipelines/svm_ecfp/

RF: /scratch/work/project/excape-public/pipelines/rf_ecfp/

4.2 Matrix factorization

Computational chemogenomics pipelines developed with SMURFF fall into two main categories: search for the best hyperparameter settings (HP search) and prediction mode. Both are available thru command line interface and Python API, that allow to deploy SMURFF thru Jupyter notebooks. Example of them are provided at:

https://github.com/ExaScience/smurff/tree/master/docs/notebooks

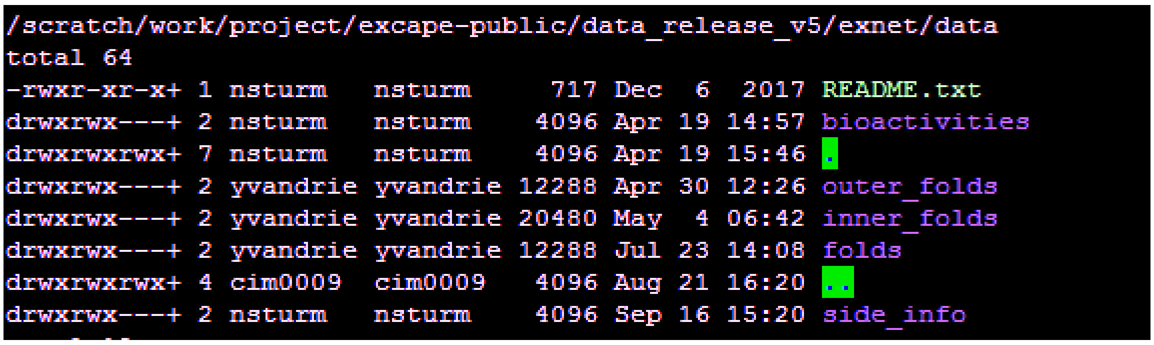
- **input_matrices_and_tensors.ipynb** – different types of the data supported by SMURFF, namely matrices and tensors.
- **centering.ipynb** – contains pipeline on how to pre-process the data to use with SMURFF
- **a_first_example.ipynb** – contains some basic example for building a model using a subset of ChEMBL, as full runs on ExCAPEDB datasets are not advisable through Jupyter notebooks.
- **different_methods.ipynb** – New notebooks for centering and input
- **different_noise_models.ipynb** New notebooks for centering and input
- **inference_with_smurff.ipynb** – example providing basic coverage of the
- **syn_out_matrix_prediction.ipynb** – example for out-of-matrix prediction for synthetic data. Common way to validate matrix factorization methods is to sample cells of the matrix for validation, while in chemogenomics common way to validate the models is out-of-matrix, for compounds that does not have any information on the Y matrix.

An example on how to run SMURFF on Salomon is provided on ExCAPE *git: documents/wp3/macau_notebooks/Run_SMURFF_on_Salomon.ipynb*

4.3 Deep learning

The large scale deep net experiments are performed with **ExNET**, available at *git:excape/ code/wp2/exnet/tf.* The data folder used is:

"/scratch/work/project/excape-public/data_release_v5/exnet/data" (Figure 5)



```
/scratch/work/project/excape-public/data_release_v5/exnet/data
total 64
-rwxr-xr-x+ 1 nsturm   nsturm     717 Dec  6  2017 README.txt
drwxrwx---+ 2 nsturm   nsturm    4096 Apr 19 14:57 bioactivities
drwxrwxrwx+ 7 nsturm   nsturm    4096 Apr 19 15:46 .
drwxrwx---+ 2 yvandrie yvandrie 12288 Apr 30 12:26 outer_folds
drwxrwx---+ 2 yvandrie yvandrie 20480 May  4 06:42 inner_folds
drwxrwx---+ 2 yvandrie yvandrie 12288 Jul 23 14:08 folds
drwxrwxrwx+ 4 cim0009  cim0009   4096 Aug 21 16:20 ..
drwxrwx---+ 2 nsturm   nsturm    4096 Sep 16 15:20 side_info
```

Figure 5. Data release with machine learning friendly formats, overview

4.3.1 ECFP Fingerprints

Training Exnet models with ECFP fingerprints was done with Exnet 3 pipeline (provided by It4i) and available at */scratch/work/project/excape-public/temp/edlp (pipeline.py)*.

Improved support for calculating custom evaluation metrics was introduced in Exnet v4, therefore Exnet.v4 pipeline was used for calculating various evaluation metrics /scratch/work/project/excape-public/temp/exnet4-vojtech-temp

Features matrix: data v5 exnet/data/side info/ECFP6 counts var005.mtx

Data directory: data release v5/exnet/data

4.3.2 Dense descriptors (chem2vec)

Training Exnet models with ECFP fingerprints was done with Exnet v4, as this is the first version supporting dense descriptors. The Exnet v4 was further modified with improved writing of result files. Evaluation script supporting multiple metrics was implemented (AUC, logloss, F1, Kappa, and confusion matrix at thresholds 0.3, 0.4, 0.5, 0.6, 0.7). The updated code is available at git: code/wp3/exnet/c2v/tf and on Salomon (see below).

The workflows for chem2vec were disseminated in: Poster "*chem2vec: vector embedding of atoms and molecules*" at ICCS 2018, May27-31, The Netherlands

Three versions of chem2vec descriptors were explored, namely chem2vec40sum, chem2vec40prod, chem2vec67prod (see D3.18 for details). The descriptors are standardized via subtracting the mean and dividing on standard deviation. The statistics are stored as txt file for further use. The pipeline script provided by IT4I was customized to make use of the new ExNET option allowing to specify output evaluation file and checks for existing evaluation files. The pipelines for different folds and chem2vec versions are identical up to configuration details. The pipelines are also available on git: code/wp3/exnet/c2v and Salomon (see below).

ExNET inner and outer folds: /scratch/work/project/excape-public/temp/exnet4/tf/exnet.py

ExNET full data: /scratch/work/project/excape-public/temp/exnet4/tf/exnet_full.py

Multi-score evaluation script:

/scratch/work/project/excape-public/temp/exnet4/tf/evaluation.py

Features matrix:

data v5 exnet/data/side info/chem2vec40sum_std.mtx (chem2vec d=40*7 , combined as sum, standardized)

data v5 exnet/data/side info/chem2vec40prod_std.mtx (chem2vec d=40*7 , combined as product, standardized)

data v5 exnet/data/side info/chem2vec_std.mtx (chem2vec d=67*7 , combined as product, standardized)

Data directory: data release v5/exnet/data

Pipelines: pipeline_exnet4.4.py at

/scratch/work/project/excape-public/temp/exnet_metrics/c2v40_prod_outer

/scratch/work/project/excape-public/temp/exnet_metrics/c2v40_prod_inner

/scratch/work/project/excape-public/temp/exnet_metrics/c2v40_prod_full

/scratch/work/project/excape-public/temp/exnet_metrics/c2v67_prod_outer

/scratch/work/project/excape-public/temp/exnet_metrics/c2v67_prod_inner

/scratch/work/project/excape-public/temp/exnet_metrics/c2v67_prod_full

Result files at /scratch/work/project/excape-public/temp/

results_data_v5_exnet_c2v40prod

results_data_v5_exnet_c2v40prod_full

results_data_v5_exnet_c2v40prod_inner

results_data_v5_exnet_c2v40sum

results_data_v5_exnet_c2v40sum_full

results_data_v5_exnet_c2v40sum_inner

results_data_v5_exnet_c2v67prod_full

results_data_v5_exnet_c2v67std_inner

results_data_v5_exnet_c2v67std_outer

results_data_v5_exnet_c2v67std_outer_long

4.3.3 Tensorflow kit

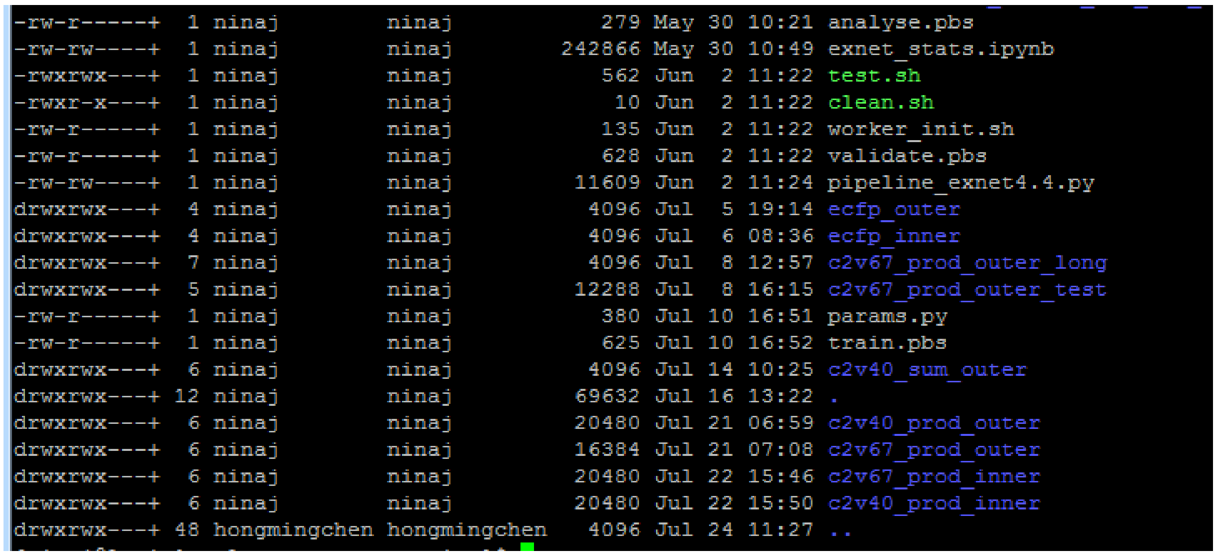
Tensorflow kit is a small collection of python scripts for running neural-net models with Tensorflow. All scripts are both Python 2.7 and Python 3.6 tested and compatible with TF versions 1.0 and 0.12. Also, all runnable scripts can be invoked with -h argument to get help information about them. The paradigm behind organizing the scripts is to separate the model creation process from training and from inference runs of the trained model. Following that idea, the two implemented types of models - (Denoising) Auto-Encoders and Variational Auto-Encoders have the scripts tf_ae.py and tf_vae.py which only purpose is to build the architecture given in the arguments. The library is released as open source by Ideaconsult and available at https://github.com/ideaconsult/tf_kit . It was used to run experiments with autoencoders in ExCAPE.

4.3.4 Analysis

Data analysis is performed with Jupyter notebooks, available at *git: code/wp3/ExcapeDB/exnet_analysis*.

- excape_models.py: all large scale ExCAPE models (exnet, xgboost, RF, SVM with ECFP and chem2vec flavours, inner and outer folds) are defined as one enum class.
- exnet_models_params.py: defines which model(s) to process, e.g. by setting _modelname = "exnet_c2v67prod_inner". This file allows to parameterize the notebooks below, by deploying the same notebook in separate folders (e.g. different methods or chem2vec flavours), with specific exnet_models_params.py settings.
- singletask_metrics.ipynb: processes evaluation results of models defined in exnet_models_params.py as e.g. _xbmodelname = "xgboost_ecfp_inner" . Creates a summary file. Finds the best models by different criteria. Outputs statistics and charts and dictionaries of best models per target (as tab delimited files). This notebook works for xgboost, random forest and SVM results.
- exnet_metrics.ipynb: processes evaluation results of models defined in exnet_models_params.py as e.g. _modelname = "exnet_c2v40prod_inner". If a summary file from is not available, reads all metrics.tsv files from Exnet runs for the specified model and generates summary.txt. The summary file is generated only once and contains performance statistics for all checkpoints and all hyperparameters explored. Then the notebook code finds the best models by different criteria. Outputs statistics and charts and dictionaries of best models per target (as tab delimited files).
- exnet_inner_vs_outer.ipynb : Uses _model_inner_name and _model_outer_name as defined in exnet_models_params.py. Processes the files generated by exnet_metrics.ipynb (inner folds processing and model selection). Generates outer fold statistics and charts.
- exnet_compare.ipynb : A notebook used to generate comparative statistics involving several different models, e.g. violin plots in D3.18 (multitask vs single task learners) or comparison between ECFP and chem2vec.

The notebooks are deployed in all folders of /scratch/work/project/excape-public/temp/exnet_metrics/ and launched via respective analyse.pbs.



```
-rw-r-----+  1 ninaj        ninaj           279 May 30 10:21 analyse.pbs
-rw-rw----+  1 ninaj        ninaj        242866 May 30 10:49 exnet_stats.ipynb
-rwxrwx---+  1 ninaj        ninaj           562 Jun  2 11:22 test.sh
-rwxr-x---+  1 ninaj        ninaj            10 Jun  2 11:22 clean.sh
-rw-r-----+  1 ninaj        ninaj           135 Jun  2 11:22 worker_init.sh
-rw-r-----+  1 ninaj        ninaj           628 Jun  2 11:22 validate.pbs
-rw-rw----+  1 ninaj        ninaj         11609 Jun  2 11:24 pipeline_exnet4.4.py
drwxrwx---+  4 ninaj        ninaj          4096 Jul  5 19:14 ecfp_outer
drwxrwx---+  4 ninaj        ninaj          4096 Jul  6 08:36 ecfp_inner
drwxrwx---+  7 ninaj        ninaj          4096 Jul  8 12:57 c2v67_prod_outer_long
drwxrwx---+  5 ninaj        ninaj         12288 Jul  8 16:15 c2v67_prod_outer_test
-rw-r-----+  1 ninaj        ninaj           380 Jul 10 16:51 params.py
-rw-r-----+  1 ninaj        ninaj           625 Jul 10 16:52 train.pbs
drwxrwx---+  6 ninaj        ninaj          4096 Jul 14 10:25 c2v40_sum_outer
drwxrwx---+ 12 ninaj        ninaj         69632 Jul 16 13:22 .
drwxrwx---+  6 ninaj        ninaj         20480 Jul 21 06:59 c2v40_prod_outer
drwxrwx---+  6 ninaj        ninaj         16384 Jul 21 07:08 c2v67_prod_outer
drwxrwx---+  6 ninaj        ninaj         20480 Jul 22 15:46 c2v67_prod_inner
drwxrwx---+  6 ninaj        ninaj         20480 Jul 22 15:50 c2v40_prod_inner
drwxrwx---+ 48 hongmingchen hongmingchen   4096 Jul 24 11:27 ..
```

Figure 6.

Notebooks provided by AZ:

Xgboost results analysis:

/scratch/work/project/excape-public/temp/xgboost_metrics/XGboost - results analysis.ipynb

Svm-linear results analysis:

/scratch/work/project/excape-public/temp/svm_metrics/precisionRecallKappa_svm.py

4.4 Conformal prediction

A notebook, calculating Venn-ABERS probabilities for compounds, using fast isotonic regression
Available at excape git: code/wp3/excapeml/calibrate_va

4.5 Single task learners

4.5.1 Xgboost (ECFP)

Feature matrix data v5 exnet/data/side info/ECFP6 counts var005.mtx

Data directory data release v5/exnet/data

Pipeline git:code/wp2/loom-pipelines/loom-xgboost

/scratch/work/project/excape-public/pipelines/xgbp_ecfp

Result files at /scratch/work/project/excape-public/temp/

results_v5_xgboost_ecfp
results_v5_xgboost_ecfp_final
results_v5_xgboost_ecfp_inner

4.5.2 SVM (ECFP)

Feature matrix data v5 exnet/data/side info/ECFP6 counts var005.mtx

Data directory data release v5/exnet/data

Pipeline git:code/wp2/loom-pipelines/loom-libsvm

scratch/work/project/excape-public/pipelines/svm_ecfp

Result files at /scratch/work/project/excape-public/temp/

results_v5_svm_ecfp

results_v5_svm_ecfp_final

4.5.3 Random forest (ECFP)

Feature matrix data v5 exnet/data/side info/ECFP6 counts var005.mtx

Data directory data release v5/exnet/data

Pipeline git:code/wp2/loom-pipelines/loom-libsvm

/scratch/work/project/excape-public/pipelines/rfp_ecfp

Result files at /scratch/work/project/excape-public/temp/

results_v5_rf_ecfp
results_v5_rf_ecfp_final

4.5.4 Xgboost (chem2vec)

Feature matrix data v5 exnet/data/side info/ chem2vec.mtx

Data directory data release v5/exnet/data

Pipeline git:code/wp2/loom-pipelines/loom-xgboost

scratch/work/project/excape-public/pipelines/xgbp_c2v

Result files at /scratch/work/project/excape-public/temp/

results_v5_xgboost_c2v
results_v5_xgboost_c2v_inner

4.5.5 SVM (chem2vec)

Feature matrix data v5 exnet/data/side info/ chem2vec_std.mtx

Data directory data release v5/exnet/data

Pipeline git:code/wp2/loom-pipelines/loom-libsvm

scratch/work/project/excape-public/pipelines/svm_c2v

Result files at /scratch/work/project/excape-public/temp/

results_v5_svm_c2v

4.5.6 Random forest (chem2vec)

Feature matrix data v5 exnet/data/side info/ chem2vec.mtx

Data directory data release v5/exnet/data

Pipeline git:code/wp2/loom-pipelines/loom-libsvm

scratch/work/project/excape-public/pipelines/rf_c2v

Result files at /scratch/work/project/excape-public/temp/

results_v5_rf_c2v

5 Conclusion

Jupyter notebooks to prepare datasets, train, evaluate and apply large scale multitask and single tasks model are developed. The pipeline code is available on ExCAPE git, with some of the tools released under open license (SMURFF, Ambit, HyperLoom, tf_kit). The type of models include deep nets (ExNET) , matrix factorization (SMURFF) , single task (XGBoost) and single task baseline models like Random forest and SVM. Pipelines for sparse (ECFP) and dense (chem2vec) descriptors are deployed and used to evaluate performance (reported in D3.18).

6 References

1. Vojtěch Cima et al., HyperLoom: A Platform for Defining and Executing Scientific Pipelines in Distributed Environments, **2018**, *Proceedings of the 9th Workshop and 7th Workshop on Parallel Programming and RunTime Management Techniques for Manycore Architectures and Design Tools and Architectures for Multicore Embedded Computing Platform, Manchester, United Kingdom*.

2. http://jupyter.org/

3. https://anaconda.org/

4. Andreas Mayr et al., DeepTox: Toxicity prediction using deep learning, **2016,** *Front. Envir. Sci., 3, 80*

5. https://www.scipy.org/