Test protocol for Image2Reg

By Daniel Paysan
October 6th,, 2023

1. Installation

Clone repository
Time: 3 minutes
Size: 2 GB

```
git clone https://github.com/uhlerlab/image2reg.git
cd image2reg
```

Create environment
Time: 1 minutes
Size: >1 GB

```
conda create --name image2reg python==3.8.10
```

```
(base) daniel@nacho:~/Desktop/image2reg$ conda create --name image2reg python=3.8.10
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.
Collecting package metadata (repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 23.1.0
  latest version: 23.7.2

Please update conda by running

    $ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

     conda install conda=23.7.2



## Package Plan ##

  environment location: /usr/share/miniconda3/envs/image2reg

  added / updated specs:
    - python=3.8.10
```

```
The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    python-3.8.10              |       h12debd9_8        57.7 MB
    ------------------------------------------------------------
                                           Total:        57.7 MB

The following NEW packages will be INSTALLED:

  _libgcc_mutex      pkgs/main/linux-64::_libgcc_mutex-0.1-main
  _openmp_mutex      pkgs/main/linux-64::_openmp_mutex-5.1-1_gnu
  ca-certificates    pkgs/main/linux-64::ca-certificates-2023.05.30-h06a4308_0
  ld_impl_linux-64   pkgs/main/linux-64::ld_impl_linux-64-2.38-h1181459_1
  libffi             pkgs/main/linux-64::libffi-3.3-he6710b0_2
  libgcc-ng          pkgs/main/linux-64::libgcc-ng-11.2.0-h1234567_1
  libgomp            pkgs/main/linux-64::libgomp-11.2.0-h1234567_1
  libstdcxx-ng       pkgs/main/linux-64::libstdcxx-ng-11.2.0-h1234567_1
  ncurses            pkgs/main/linux-64::ncurses-6.4-h6a678d5_0
  openssl            pkgs/main/linux-64::openssl-1.1.1v-h7f8727e_0
  pip                pkgs/main/linux-64::pip-23.2.1-py38h06a4308_0
  python             pkgs/main/linux-64::python-3.8.10-h12debd9_8
  readline           pkgs/main/linux-64::readline-8.2-h5eee18b_0
  setuptools         pkgs/main/linux-64::setuptools-68.0.0-py38h06a4308_0
  sqlite             pkgs/main/linux-64::sqlite-3.41.2-h5eee18b_0
  tk                 pkgs/main/linux-64::tk-8.6.12-h1ccaba5_0
  wheel              pkgs/main/linux-64::wheel-0.38.4-py38h06a4308_0
  xz                 pkgs/main/linux-64::xz-5.4.2-h5eee18b_0
  zlib               pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0


Proceed ([y]/n)? y
```

```
Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate image2reg
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

Install dependencies

Time: 15 minutes

Size: 8 GB

```
conda activate image2reg
bash scripts/installation/setup_environment_cuda.sh
```

```
(base) daniel@nacho:~/Desktop/image2reg$ conda activate image2reg
(image2reg) daniel@nacho:~/Desktop/image2reg$ bash scripts/installation/setup_environment_cuda.sh
Collecting setuptools==49.6.0 (from -r requirements/cuda/requirements_cuda1.txt (line 1))
  Downloading setuptools-49.6.0-py3-none-any.whl (803 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 803.3/803.3 kB 6.0 MB/s eta 0:00:00
Installing collected packages: setuptools
  Attempting uninstall: setuptools
    Found existing installation: setuptools 68.0.0
    Uninstalling setuptools-68.0.0:
      Successfully uninstalled setuptools-68.0.0
Successfully installed setuptools-49.6.0
Looking in links: https://download.pytorch.org/whl/torch_stable.html
Collecting nmco@ git+https://github.com/GVS-Lab/chrometrics.git@ce86d3787c661e8a59d26fcf4498efb8fd5ecc5a (fr
om -r requirements/cuda/requirements_cuda2.txt (line 118))
  Cloning https://github.com/GVS-Lab/chrometrics.git (to revision ce86d3787c661e8a59d26fcf4498efb8fd5ecc5a)
to /tmp/pip-install-a6hc4i3g/nmco_5dc59c59d3294b1ca1a2183fe739d82e
  Running command git clone --filter=blob:none --quiet https://github.com/GVS-Lab/chrometrics.git /tmp/pip-i
nstall-a6hc4i3g/nmco_5dc59c59d3294b1ca1a2183fe739d82e
  Running command git rev-parse -q --verify 'sha^ce86d3787c661e8a59d26fcf4498efb8fd5ecc5a'
  Running command git fetch -q https://github.com/GVS-Lab/chrometrics.git ce86d3787c661e8a59d26fcf4498efb8fd
5ecc5a
  Running command git checkout -q ce86d3787c661e8a59d26fcf4498efb8fd5ecc5a
  Resolved https://github.com/GVS-Lab/chrometrics.git to commit ce86d3787c661e8a59d26fcf4498efb8fd5ecc5a
  Preparing metadata (setup.py) ... done
Collecting angel-cd==1.0.3 (from -r requirements/cuda/requirements_cuda2.txt (line 1))
  Downloading angel_cd-1.0.3-py3-none-any.whl (10 kB)
Collecting anndata==0.7.6 (from -r requirements/cuda/requirements_cuda2.txt (line 2))
  Downloading anndata-0.7.6-py3-none-any.whl (127 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.6/127.6 kB 2.0 MB/s eta 0:00:00
Collecting appdirs==1.4.4 (from -r requirements/cuda/requirements_cuda2.txt (line 3))
  Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Collecting argon2-cffi==20.1.0 (from -r requirements/cuda/requirements_cuda2.txt (line 4))
  Downloading argon2_cffi-20.1.0-cp35-abi3-manylinux1_x86_64.whl (97 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 98.0/98.0 kB 9.5 MB/s eta 0:00:00
Collecting async-generator==1.10 (from -r requirements/cuda/requirements_cuda2.txt (line 5))
  Downloading async_generator-1.10-py3-none-any.whl (18 kB)
```

```
Successfully built autograd bioservices bokeh coclust datashape easydev fdasrsf findiff future goenrich GPy
hdbscan kneebow locket loess matplotlib-venn nb-black node2vec olefile pandocfilters paramz plotbin pycluste
ring pynndescent pyrsistent python-Levenshtein python-louvain sinfo sklearn umap-learn venn nmco
DEPRECATION: nb-black 1.0.7 has a non-standard dependency specifier black>='19.3'; python_version >= "3.6".
pip 23.3 will enforce this behaviour change. A possible replacement is to upgrade to a newer version of nb-b
lack or contact the author to suggest that they release a version with a conforming dependency specifiers. D
iscussion can be found at https://github.com/pypa/pip/issues/12063
Installing collected packages: webencodings, wcwidth, typing-extensions, texttable, stdlib-list, sortedconta
iners, Send2Trash, regex, QtPy, pytz, PyQt5-sip, PuLP, ptyprocess, pickleshare, pandocfilters, nose, nmco, m
ypy-extensions, msgpack, mpmath, mistune, locket, ipython-genutils, iniconfig, HeapDict, googledrivedownload
er, certifi, backcall, appdirs, zope.interface, zope.event, zipp, zict, xmltodict, xlrd, wrapt, wheel, urlli
b3, traitlets, tqdm, tornado, toolz, tomli, toml, threadpoolctl, testpath, tblib, sympy, suds-community, sta
tannot, soupsieve, smart-open, six, sinfo, pyzmq, PyYAML, python-Levenshtein, python-igraph, PySocks, pyrsis
tent, PyQt5, pyparsing, Pygments, pycparser, pybind11, py, psutil, prompt-toolkit, prometheus-client, pluggy
, pip, Pillow, pexpect, pathspec, parso, param, olefile, numpy, networkx, nest-asyncio, natsort, multimethod
, MarkupSafe, Markdown, lxml, llvmlite, kiwisolver, jupyterlab-widgets, joblib, itsdangerous, install, idna,
 greenlet, future, fsspec, entrypoints, defusedxml, decorator, Cython, colorlog, colorama, cloudpickle, clic
k, chardet, backports.functools-lru-cache, attrs, async-generator, yacs, url-normalize, torch, thresholdclus
tering, terminado, scipy, scikit-misc, requests, PyWavelets, pyviz-comms, python-louvain, python-dateutil, P
yQtWebEngine, PyQtChart, pyct, pcst-fast, patsy, partd, packaging, opencv-python, numexpr, numba, multipledi
spatch, matplotlib-inline, mahotas, louvain, leidenalg, jupyterlab-pygments, jupyter-core, jsonschema, Jinja
2, jedi, isodate, importlib-metadata, imageio, imagecodecs-lite, h5py, gevent, fastcluster, eva-lcd, easydev
, dynetx, demon, cytoolz, cycler, chinese-whispers, cffi, black, beautifulsoup4, autograd, angel-cd, torchvi
sion, torchaudio, tifffile, tables, scikit-learn, requests-cache, rdflib, pytest, pymanopt, POT, pooch, para
mz, pandas, nbformat, matplotlib, jupyter-client, ipython, grequests, gensim, findiff, dcor, datashape, dask
, cryptography, colorcet, brotlipy, bokeh, bleach, biothings-client, argon2-cffi, yellowbrick, xarray, venn,
 statsmodels, spherecluster, sklearn, seaborn, scikit-image, scikit-datasets, random-fourier-features-pytorc
h, pyOpenSSL, pynndescent, pyclustering, pybiomart, plotbin, panel, node2vec, nf1, nbclient, nb-black, mygen
e, matplotlib-venn, markov-clustering, kneed, kneebow, ipykernel, imbalanced-learn, hdbscan, GPy, goenrich,
distributed, coclust, cmapPy, bioservices, bimlpa, axial, anndata, umap-learn, rdata, qtconsole, nbconvert,
loess, jupyter-console, holoviews, gseapy, fdasrsf, cdlib, scanpy, notebook, datashader, widgetsnbextension,
 scikit-fda, ipywidgets, jupyter
  Attempting uninstall: wheel
```

```
  Attempting uninstall: pip
    Found existing installation: pip 23.2.1
    Uninstalling pip-23.2.1:
      Successfully uninstalled pip-23.2.1
Successfully installed Cython-0.29.24 GPy-1.10.0 HeapDict-1.0.1 Jinja2-3.0.0 Markdown-3.3.4 MarkupSafe-2.0.0
 POT-0.8.1.0 Pillow-8.2.0 PuLP-2.6.0 PyQt5-5.12.3 PyQt5-sip-4.19.18 PyQtChart-5.12.0 PyQtWebEngine-5.12.1 Py
Sockets-1.7.1 PyWavelets-1.1.1 PyYAML-5.4.1 Pygments-2.9.0 QtPy-1.9.0 Send2Trash-1.5.0 angel-cd-1.0.3 anndata-
0.7.6 appdirs-1.4.4 argon2-cffi-20.1.0 async-generator-1.10 attrs-21.2.0 autograd-1.3 axial-0.2.3 backcall-0
.2.0 backports.functools-lru-cache-1.6.4 beautifulsoup4-4.11.1 bimlpa-0.1.2 bioservices-1.9.0 biothings-clie
nt-0.2.6 black-21.5b1 bleach-3.3.0 bokeh-2.3.3 brotlipy-0.7.0 cdlib-0.2.6 certifi-2021.5.30 cffi-1.14.5 char
det-4.0.0 chinese-whispers-0.8.0 click-8.0.0 cloudpickle-1.6.0 cmapPy-4.0.1 coclust-0.2.1 colorama-0.4.4 col
orcet-2.0.6 colorlog-6.6.0 cryptography-3.4.7 cycler-0.10.0 cytoolz-0.11.0 dask-2021.5.0 datashader-0.13.0 d
atashape-0.5.2 dcor-0.5.3 decorator-5.0.9 defusedxml-0.7.1 demon-2.0.6 distributed-2021.5.0 dynetx-0.3.1 eas
ydev-0.12.0 entrypoints-0.3 eva-lcd-0.1.1 fastcluster-1.2.4 fdasrsf-2.3.10 findiff-0.8.9 fsspec-2021.5.0 fut
ure-0.18.2 gensim-4.1.2 gevent-21.12.0 goenrich-1.12 googledrivedownloader-0.4 greenlet-1.1.2 grequests-0.6.
0 gseapy-0.10.6 h5py-3.2.1 hdbscan-0.8.33 holoviews-1.14.5 idna-2.10 imagecodecs-lite-2019.12.3 imageio-2.9.
0 imbalanced-learn-0.8.0 importlib-metadata-4.0.1 iniconfig-1.1.1 install-1.3.4 ipykernel-5.5.5 ipython-7.23
.1 ipython-genutils-0.2.0 ipywidgets-7.6.3 isodate-0.6.0 itsdangerous-2.0.1 jedi-0.18.0 joblib-1.0.1 jsonsch
ema-3.2.0 jupyter-1.0.0 jupyter-client-6.1.12 jupyter-console-6.4.0 jupyter-core-4.7.1 jupyterlab-pygments-0
.1.2 jupyterlab-widgets-1.0.0 kiwisolver-1.3.1 kneebow-0.1.1 kneed-0.7.0 leidenalg-0.8.7 llvmlite-0.36.0 loc
ket-0.2.0 loess-2.1.1 louvain-0.7.0 lxml-4.9.0 mahotas-1.4.11 markov-clustering-0.0.6.dev0 matplotlib-3.4.2
matplotlib-inline-0.1.2 matplotlib-venn-0.11.6 mistune-0.8.4 mpmath-1.2.1 msgpack-1.0.2 multimethod-1.7 mult
ipledispatch-0.6.0 mygene-3.2.2 mypy-extensions-0.4.3 natsort-7.1.1 nb-black-1.0.7 nbclient-0.5.3 nbconvert-
6.0.7 nbformat-5.1.3 nest-asyncio-1.5.1 networkx-2.7.1 nf1-0.0.4 nmco-0.1 node2vec-0.4.3 nose-1.3.7 notebook
-6.4.0 numba-0.53.1 numexpr-2.7.3 numpy-1.22.3 olefile-0.46 opencv-python-4.5.2.52 packaging-20.9 pandas-1.5
.1 pandocfilters-1.4.2 panel-0.12.1 param-1.11.1 paramz-0.9.5 parso-0.8.2 partd-1.2.0 pathspec-0.8.1 patsy-0
.5.1 pcst-fast-1.0.7 pexpect-4.8.0 pickleshare-0.7.5 pip-21.1.1 plotbin-3.1.3 pluggy-1.0.0 pooch-1.3.0 prome
theus-client-0.10.1 prompt-toolkit-3.0.18 psutil-5.8.0 ptyprocess-0.7.0 py-1.11.0 pyOpenSSL-20.0.1 pybind11-
2.7.0 pybiomart-0.2.0 pyclustering-0.10.1.2 pycparser-2.20 pyct-0.4.8 pymanopt-0.2.5 pynndescent-0.5.2 pypar
sing-2.4.7 pyrsistent-0.17.3 pytest-7.1.1 python-Levenshtein-0.12.2 python-dateutil-2.8.1 python-igraph-0.9.
6 python-louvain-0.16 pytz-2021.1 pyviz-comms-2.1.0 pyzmq-22.0.3 qtconsole-5.1.0 random-fourier-features-pyt
orch-1.0.0 rdata-0.7 rdflib-6.0.0 regex-2021.4.4 requests-2.25.1 requests-cache-0.7.1 scanpy-1.8.1 scikit-da
tasets-0.2.0 scikit-fda-0.7.1 scikit-image-0.18.1 scikit-learn-1.0.2 scikit-misc-0.1.4 scipy-1.8.0 seaborn-0
.11.1 sinfo-0.3.4 six-1.16.0 sklearn-0.0 smart-open-5.2.1 sortedcontainers-2.4.0 soupsieve-2.3.2.post1 spher
ecluster-0.1.7 statannot-0.2.3 statsmodels-0.12.2 stdlib-list-0.8.0 suds-community-1.1.1 sympy-1.10.1 tables
-3.6.1 tblib-1.7.0 terminado-0.9.5 testpath-0.5.0 texttable-1.6.4 threadpoolctl-2.1.0 thresholdclustering-1.
1 tifffile-2019.7.26.2 toml-0.10.2 tomli-2.0.1 toolz-0.11.1 torch-1.8.1+cu111 torchaudio-0.8.1 torchvision-0
.9.1+cu111 tornado-6.1 tqdm-4.62.2 traitlets-5.0.5 typing-extensions-3.7.4.3 umap-learn-0.5.1 url-normalize-
1.4.3 urllib3-1.26.4 venn-0.1.3 wcwidth-0.2.5 webencodings-0.5.1 wheel-0.36.2 widgetsnbextension-3.5.1 wrapt
-1.14.1 xarray-0.19.0 xlrd-1.2.0 xmltodict-0.13.0 yacs-0.1.8 yellowbrick-1.4 zict-2.0.0 zipp-3.4.1 zope.even
t-4.5.0 zope.interface-5.4.0
Looking in links: https://pytorch-geometric.com/whl/torch-1.8.1%2Bcu111.html
Collecting torch-cluster==1.5.9
  Downloading https://data.pyg.org/whl/torch-1.8.0%2Bcu111/torch_cluster-1.5.9-cp38-cp38-linux_x86_64.whl (1
.7 MB)
```

```
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/share/miniconda3/envs/image2reg/lib/python3.8/si
te-packages (from scikit-learn->torch-geometric==2.0.1->-r requirements/cuda/requirements_cuda3.txt (line 3)
) (2.1.0)
Building wheels for collected packages: torch-geometric
  Building wheel for torch-geometric (setup.py) ... done
  Created wheel for torch-geometric: filename=torch_geometric-2.0.1-py3-none-any.whl size=513805 sha256=a341
995480aa9f58a846a1d96e24f0297fcb6def77076457a7602076f26d0926
  Stored in directory: /home/daniel/.cache/pip/wheels/00/17/66/26898cf2ae68e44eeac899efb089e6a4bce7dc6a80582
2009c
Successfully built torch-geometric
Installing collected packages: torch-spline-conv, torch-sparse, torch-scatter, torch-geometric, torch-cluste
r
Successfully installed torch-cluster-1.5.9 torch-geometric-2.0.1 torch-scatter-2.0.8 torch-sparse-0.6.12 tor
ch-spline-conv-1.2.1
WARNING: You are using pip version 21.1.1; however, version 23.2.1 is available.
You should consider upgrading via the '/usr/share/miniconda3/envs/image2reg/bin/python -m pip install --upgr
ade pip' command.
(image2reg) daniel@nacho:~/Desktop/image2reg$
```

Data retrieval

Rohban data

Download data via download script

Time: 15 hours

Size: 270 GB

```
bash scripts/data/download_rohban_data.sh
```

```
(base) daniel@nacho:~/Desktop/image2reg$ conda activate image2reg
(image2reg) daniel@nacho:~/Desktop/image2reg$ bash scripts/data/download_rohban_data.sh
Starting download of the data from Rohban et al. (2017): IDR0033
Downloading ssh key...
--2023-08-05 23:34:11--  https://idr.openmicroscopy.org/about/img/aspera/asperaweb_id_dsa.openssh
Resolving idr.openmicroscopy.org (idr.openmicroscopy.org)... 45.88.81.28
Connecting to idr.openmicroscopy.org (idr.openmicroscopy.org)|45.88.81.28|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 668 [application/octet-stream]
Saving to: ‘asperaweb_id_dsa.openssh’

asperaweb_id_dsa.openssh   100%[=====================================>]     668  --.-KB/s    in 0s

2023-08-05 23:34:11 (316 MB/s) - ‘asperaweb_id_dsa.openssh’ saved [668/668]

Starting download...
Downloading imaging data...
taoe005-u2os-72h-cp-a-au00044858_a01_s1_w1b6c                 100% 2286KB 10.9Mb/s    00:01
taoe005-u2os-72h-cp-a-au00044858_a01_s1_w28d3                 100% 2286KB 29.6Mb/s    00:01
taoe005-u2os-72h-cp-a-au00044858_a01_s1_w39da                 100% 2286KB 37.2Mb/s    00:02
taoe005-u2os-72h-cp-a-au00044858_a01_s1_w4fa3                 100% 2286KB 37.1Mb/s    00:02
taoe005-u2os-72h-cp-a-au00044858_a01_s2_w1e16                 100% 2286KB 36.1Mb/s    00:03
taoe005-u2os-72h-cp-a-au00044858_a01_s1_w5078                 100% 2286KB 36.8Mb/s    00:04
taoe005-u2os-72h-cp-a-au00044858_a01_s2_w2a6b                 100% 2286KB 36.9Mb/s    00:04
taoe005-u2os-72h-cp-a-au00044858_a01_s2_w4dca                 100% 2286KB 36.5Mb/s    00:05
taoe005-u2os-72h-cp-a-au00044858_a01_s2_w397a                 100% 2286KB 36.9Mb/s    00:05
taoe005-u2os-72h-cp-a-au00044858_a01_s2_w5be0                 100% 2286KB 35.5Mb/s    00:06
taoe005-u2os-72h-cp-a-au00044858_a01_s3_w15a7                 100% 2286KB 35.7Mb/s    00:06
taoe005-u2os-72h-cp-a-au00044858_a01_s3_w25d0                 100% 2286KB 35.6Mb/s    00:07
taoe005-u2os-72h-cp-a-au00044858_a01_s3_w3e6c                 100% 2286KB 36.6Mb/s    00:07
taoe005-u2os-72h-cp-a-au00044858_a01_s3_w4fee                 100% 2286KB 36.8Mb/s    00:08
taoe005-u2os-72h-cp-a-au00044858_a01_s3_w5ffa                 100% 2286KB 36.7Mb/s    00:08
taoe005-u2os-72h-cp-a-au00044858_a01_s4_w1af5                 100% 2286KB 36.4Mb/s    00:09
taoe005-u2os-72h-cp-a-au00044858_a01_s4_w2f76                 100% 2286KB 35.7Mb/s    00:09
taoe005-u2os-72h-cp-a-au00044858_a01_s4_w3d9b                 100% 2286KB 36.8Mb/s    00:10
taoe005-u2os-72h-cp-a-au00044858_a01_s4_w4d50                 100% 2286KB 36.6Mb/s    00:10
```

```
taoe005-u2os-72h-cp-a-au00044859_p24_s7_w2df6 100% 2286KB 37.2Mb/s 14:08:38
taoe005-u2os-72h-cp-a-au00044859_p24_s7_w3cfa 100% 2286KB 39.0Mb/s 14:08:39
taoe005-u2os-72h-cp-a-au00044859_p24_s7_w4009 100% 2286KB 36.4Mb/s 14:08:39
taoe005-u2os-72h-cp-a-au00044859_p24_s7_w5851 100% 2286KB 36.4Mb/s 14:08:40
taoe005-u2os-72h-cp-a-au00044859_p24_s8_w176e 100% 2286KB 39.1Mb/s 14:08:40
taoe005-u2os-72h-cp-a-au00044859_p24_s8_w2e03 100% 2286KB 35.5Mb/s 14:08:41
taoe005-u2os-72h-cp-a-au00044859_p24_s8_w3772 100% 2286KB 35.5Mb/s 14:08:41
taoe005-u2os-72h-cp-a-au00044859_p24_s8_w443c 100% 2286KB 38.1Mb/s 14:08:42
taoe005-u2os-72h-cp-a-au00044859_p24_s8_w5921 100% 2286KB 37.3Mb/s 14:08:42
taoe005-u2os-72h-cp-a-au00044859_p24_s9_w1125 100% 2286KB 37.3Mb/s 14:08:43
taoe005-u2os-72h-cp-a-au00044859_p24_s9_w2af2 100% 2286KB 38.3Mb/s 14:08:43
taoe005-u2os-72h-cp-a-au00044859_p24_s9_w3845 100% 2286KB 38.3Mb/s 14:08:44
taoe005-u2os-72h-cp-a-au00044859_p24_s9_w5c05 100% 2286KB 38.2Mb/s 14:08:44
taoe005-u2os-72h-cp-a-au00044859_p24_s9_w4992 100% 2286KB 38.2Mb/s 14:08:45
Completed: 237094913K bytes transferred in 50925 seconds
 (38139K bits/sec), in 103684 files, 7 directories.
Downloading metadata...
TargetAccelerator.sql                         100%   23GB 38.8Mb/s  1:26:21
Per_Object_View.sql                           100%  191KB 38.8Mb/s  1:26:21
Completed: 24541020K bytes transferred in 5181 seconds
 (38802K bits/sec), in 2 files, 1 directory.
Data stored in raw/data
Exiting...
(image2reg) daniel@nacho:~/Desktop/image2reg$
```

Clean data download and extract metadata from database files

Time: 45 minutes

Size: 40 GB

```
conda activate image2reg
bash scripts/data/prepare_rohban_data.sh
```

```
(base) daniel@nacho:/media/daniel/T7/image2reg$ conda activate image2reg
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ bash scripts/data/prepare_rohban_data.sh
Enter password:
Enter password:
Enter password:
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$
```

GEX data

Download CMap gene signatures
Time: 10 minutes
Size: 3 GB

- Create the directory to store the CMap data via:

```
mkdir -p data/resources/gex/cmap
```

- Download the following files from https://clue.io/data/CMap2020#LINCS2020
    - cellinfo_beta.txt
    - geneinfo_beta.txt
    - siginfo_beta.txt
    - Level5_beta_trt_oe_n34171x12328.gctx
- Place the files in the created “cmap” directory

```
(base) daniel@nacho:/media/daniel/T7/image2reg$ conda activate image2reg
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ mkdir -p data/resources/gex/cmap
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ cd data/resources/gex/cmap
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/cmap$ ls
cellinfo_beta.txt  instinfo_beta.txt                  siginfo_beta.txt
geneinfo_beta.txt  level5_beta_trt_oe_n34171x12328.gctx
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/cmap$
```

Download scRNA-seq from the GEO database
Time: > 1 minute
Size: > 1 GB

- Create the directory to store the scRNA-seq data

```
mkdir -p data/resources/gex/scrnaseq
```

- Download the file GSE146773_Counts.csv.gz from https://www.ncbi.nlm.nih.gov/geo/download/?acc=GSE146773&format=file&file=GSE146773%5FCounts%2Ecsv%2Egz
- Place it into the created scRNA-seq directory
- Unzip the read count file

```
find . -name '*.csv.gz' -print0 | xargs -0 -n1 gzip -d
```

```
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/scrnaseq$ ls
GSE146773_Counts.csv.gz
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/scrnaseq$ find .
 -name '*.csv.gz' -print0 | xargs -0 -n1 gzip -d
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/scrnaseq$ ls
GSE146773_Counts.csv
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/scrnaseq$
```

Download bulk RNAseq from the CCLE database

Time: > 1 minute

Size: > 1 GB

- Create the directory to store the bulk RNA-seq data

```
mkdir -p data/resources/gex/ccle
```

- Download the following files from https://depmap.org/portal/download/all/ . Make sure to select the DepMap version 21Q2.
    - CCLE_expression.csv
    - sample_info.csv
- Place the two files in the created directory for the (CCLE) bulk RNA-seq data
- Rename the sample_info.csv file to CCLE_expression_sample_info.csv for better association

```
cd data/resources/gex/ccle
mv sample_info.csv CCLE_expression_sample_info.csv
```

```
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data$ cd ..
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ mkdir -p data/resources/gex/ccle
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ cd data/resources/gex/ccle
mv sample_info.csv CCLE_expression_sample_info.csv
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/ccle$ ls
CCLE_expression.csv  CCLE_expression_sample_info.csv
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/gex/ccle$
```

Gene set information are enclosed in the github repository

Time: 1 minute

Size: > 1 GB

Gene set information was obtained from multiple sources as described in the manuscript. Since these gene sets are subject to change, we have provided the respective data as part of our github repository to reproduce our results.

- Create a directory to store the gene set information

```
mkdir -p data/resources/genesets
```

- Move the gene set information from other/genesets to the created directory

```
mv other/genesets data/resources/genesets
```

```
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ mkdir -p data/resources/genesets
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ cd data/resources/genesets/
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/genesets$ ls
c2.cp.biocarta.v7.5.1.symbols.gmt      l1000.txt
c2.cp.kegg.v7.5.1.symbols.gmt          protein_coding_gene_list.txt
c2.cp.pid.v7.5.1.symbols.gmt           reactome_cell_cycle.txt
c2.cp.wikipathways.v7.5.1.symbols.gmt  reactome_cell_death.txt
h.all.v7.4.symbols.gmt                 reactome_chrom_org.txt
human_tf_list.txt                      reactome_dna_repair.txt
kegg_reg_act_cytoskeleton.txt
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/genesets$
```

Protein-protein interaction data

Download the iRefIndexDB v14 data from the OmicsIntegrator2 github
Time: 1 min
Size: 1 GB

- Create the directory to store the data of the human protein-protein interactome as provided by the iRefIndexDB v14.

```
mkdir -p data/resources/ppi
```

- Download the preprocessed version of the interactome from the OmicsIntegrator2 Github repository available at: https://github.com/fraenkel-lab/OmicsIntegrator2/tree/master

```
cd data/resources/ppi
wget
"https://raw.githubusercontent.com/fraenkel-lab/OmicsIntegrator2/master/
example/OI2_pipeline_data/iRefIndex_v14_MIScore_interactome_C9.costs.txt
"
```

```
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ mkdir -p data/resources/ppi
(image2reg) daniel@nacho:/media/daniel/T7/image2reg$ cd data/resources/ppi
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/ppi$ wget "https://raw.githubuserc
ontent.com/fraenkel-lab/OmicsIntegrator2/master/example/OI2_pipeline_data/iRefIndex_v14_MIScore_inter
actome_C9.costs.txt"
--2023-08-07 14:09:13--  https://raw.githubusercontent.com/fraenkel-lab/OmicsIntegrator2/master/examp
le/OI2_pipeline_data/iRefIndex_v14_MIScore_interactome_C9.costs.txt
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.110.133,
185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 3814660 (3.6M) [text/plain]
Saving to: ‘iRefIndex_v14_MIScore_interactome_C9.costs.txt’

iRefIndex_v14_MIScore_int 100%[===================================>]   3.64M  9.47MB/s    in 0.4s

2023-08-07 14:09:14 (9.47 MB/s) - ‘iRefIndex_v14_MIScore_interactome_C9.costs.txt’ saved [3814660/381
4660]

(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/ppi$ ls
iRefIndex_v14_MIScore_interactome_C9.costs.txt
(image2reg) daniel@nacho:/media/daniel/T7/image2reg/data/resources/ppi$
```

Data preprocessing

Rohban data

Setup environment for nuclear segmentation using a pretrained UNet model from volkerh/unet

Time: 5 minutes

Size: 5 GB

- Clone the repository that contains the code base for the UNet-based segmentation model

```
git clone "https://github.com/dpaysan/unet-nuclei.git"
```

```
(base) daniel@nacho:/media/daniel/T7$ git clone "https://github.com/dpaysan/unet-nuclei.git"
Cloning into 'unet-nuclei'...
remote: Enumerating objects: 173, done.
remote: Counting objects: 100% (51/51), done.
remote: Compressing objects: 100% (45/45), done.
remote: Total 173 (delta 6), reused 49 (delta 5), pack-reused 122
Receiving objects: 100% (173/173), 82.30 MiB | 12.45 MiB/s, done.
Resolving deltas: 100% (53/53), done.
(base) daniel@nacho:/media/daniel/T7$ cd unet-nuclei/
(base) daniel@nacho:/media/daniel/T7/unet-nuclei$ ls
data         MANIFEST.in  readme_image.png  requirements.txt  src    unet_nuclei
LICENSE.md   notebook     README.md         setup.py          tests
(base) daniel@nacho:/media/daniel/T7/unet-nuclei$
```

- Create the conda environment containing the dependencies used to run the UNet segmentation

```
conda create --name unet python=3.8.10
```

```
(base) daniel@nacho:/media/daniel/T7/unet-nuclei$ conda create --name unet python=3.8.10
Collecting package metadata (current_repodata.json): done
Solving environment: failed with repodata from current_repodata.json, will retry with next repodata s
ource.
Collecting package metadata (repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 23.1.0
  latest version: 23.7.2

Please update conda by running

    $ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

     conda install conda=23.7.2



## Package Plan ##

  environment location: /usr/share/miniconda3/envs/unet

  added / updated specs:
    - python=3.8.10
```

```
The following NEW packages will be INSTALLED:

  _libgcc_mutex      pkgs/main/linux-64::_libgcc_mutex-0.1-main
  _openmp_mutex      pkgs/main/linux-64::_openmp_mutex-5.1-1_gnu
  ca-certificates    pkgs/main/linux-64::ca-certificates-2023.05.30-h06a4308_0
  ld_impl_linux-64   pkgs/main/linux-64::ld_impl_linux-64-2.38-h1181459_1
  libffi             pkgs/main/linux-64::libffi-3.3-he6710b0_2
  libgcc-ng          pkgs/main/linux-64::libgcc-ng-11.2.0-h1234567_1
  libgomp            pkgs/main/linux-64::libgomp-11.2.0-h1234567_1
  libstdcxx-ng       pkgs/main/linux-64::libstdcxx-ng-11.2.0-h1234567_1
  ncurses            pkgs/main/linux-64::ncurses-6.4-h6a678d5_0
  openssl            pkgs/main/linux-64::openssl-1.1.1v-h7f8727e_0
  pip                pkgs/main/linux-64::pip-23.2.1-py38h06a4308_0
  python             pkgs/main/linux-64::python-3.8.10-h12debd9_8
  readline           pkgs/main/linux-64::readline-8.2-h5eee18b_0
  setuptools         pkgs/main/linux-64::setuptools-68.0.0-py38h06a4308_0
  sqlite             pkgs/main/linux-64::sqlite-3.41.2-h5eee18b_0
  tk                 pkgs/main/linux-64::tk-8.6.12-h1ccaba5_0
  wheel              pkgs/main/linux-64::wheel-0.38.4-py38h06a4308_0
  xz                 pkgs/main/linux-64::xz-5.4.2-h5eee18b_0
  zlib               pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0


Proceed ([y]/n)? y


Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate unet
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) daniel@nacho:/media/daniel/T7/unet-nuclei$
```

- Activate conda environment and install require software libraries

```
conda activate unet
bash setup_unet_environment.sh
```

```
(base) daniel@nacho:/media/daniel/T7/unet-nuclei$ conda activate unet
(unet) daniel@nacho:/media/daniel/T7/unet-nuclei$ bash setup_unet_environment.sh
Collecting setuptools==49.6.0
  Downloading setuptools-49.6.0-py3-none-any.whl (803 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 803.3/803.3 kB 8.7 MB/s eta 0:00:00
Installing collected packages: setuptools
  Attempting uninstall: setuptools
    Found existing installation: setuptools 68.0.0
    Uninstalling setuptools-68.0.0:
      Successfully uninstalled setuptools-68.0.0
Successfully installed setuptools-49.6.0
Collecting Keras==2.4.3
  Downloading Keras-2.4.3-py2.py3-none-any.whl (36 kB)
```

```
  Attempting uninstall: h5py
    Found existing installation: h5py 3.9.0
    Uninstalling h5py-3.9.0:
      Successfully uninstalled h5py-3.9.0
Successfully installed MarkupSafe-2.1.3 absl-py-0.15.0 astunparse-1.6.3 cachetools-5.3.1 certifi-2023
.7.22 charset-normalizer-3.2.0 flatbuffers-1.12 gast-0.4.0 google-auth-2.22.0 google-auth-oauthlib-0.
4.6 google-pasta-0.2.0 grpcio-1.34.1 h5py-3.1.0 idna-3.4 importlib-metadata-6.8.0 keras-nightly-2.5.0
.dev2021032900 keras-preprocessing-1.1.2 markdown-3.4.4 oauthlib-3.2.2 opt-einsum-3.3.0 protobuf-3.20
.3 pyasn1-0.5.0 pyasn1-modules-0.3.0 requests-2.31.0 requests-oauthlib-1.3.1 rsa-4.9 six-1.15.0 tenso
rboard-2.11.2 tensorboard-data-server-0.6.1 tensorboard-plugin-wit-1.8.1 tensorflow-2.5.0 tensorflow-
estimator-2.5.0 termcolor-1.1.0 typing-extensions-3.7.4.3 urllib3-1.26.16 werkzeug-2.3.6 wrapt-1.12.1
 zipp-3.16.2
Collecting jupyter==1.0.0
```

```
  Attempting uninstall: typing-extensions
    Found existing installation: typing-extensions 3.7.4.3
    Uninstalling typing-extensions-3.7.4.3:
      Successfully uninstalled typing-extensions-3.7.4.3
ERROR: pip's dependency resolver does not currently take into account all the packages that are insta
lled. This behaviour is the source of the following dependency conflicts.
tensorflow 2.5.0 requires typing-extensions~=3.7.4, but you have typing-extensions 4.7.1 which is inc
ompatible.
Successfully installed anyio-3.7.1 argon2-cffi-21.3.0 argon2-cffi-bindings-21.2.0 arrow-1.2.3 asttoke
ns-2.2.1 async-lru-2.0.4 attrs-23.1.0 babel-2.12.1 backcall-0.2.0 beautifulsoup4-4.12.2 bleach-6.0.0
cffi-1.15.1 comm-0.1.4 debugpy-1.6.7 decorator-5.1.1 defusedxml-0.7.1 exceptiongroup-1.1.2 executing-
1.2.0 fastjsonschema-2.18.0 fqdn-1.5.1 importlib-resources-6.0.1 ipykernel-6.25.0 ipython-8.12.2 ipyt
hon-genutils-0.2.0 ipywidgets-8.1.0 isoduration-20.11.0 jedi-0.19.0 jinja2-3.1.2 json5-0.9.14 jsonpoi
nter-2.4 jsonschema-4.19.0 jsonschema-specifications-2023.7.1 jupyter-1.0.0 jupyter-client-8.3.0 jupy
ter-console-6.6.3 jupyter-core-5.3.1 jupyter-events-0.7.0 jupyter-lsp-2.2.0 jupyter-server-2.7.0 jupy
ter-server-terminals-0.4.4 jupyterlab-4.0.4 jupyterlab-pygments-0.2.2 jupyterlab-server-2.24.0 jupyte
rlab-widgets-3.0.8 matplotlib-inline-0.1.6 mistune-3.0.1 nbclient-0.8.0 nbconvert-7.7.3 nbformat-5.9.
2 nest-asyncio-1.5.7 notebook-7.0.2 notebook-shim-0.2.3 overrides-7.4.0 packaging-23.1 pandocfilters-
1.5.0 parso-0.8.3 pexpect-4.8.0 pickleshare-0.7.5 pkgutil-resolve-name-1.3.10 platformdirs-3.10.0 pro
metheus-client-0.17.1 prompt-toolkit-3.0.39 psutil-5.9.5 ptyprocess-0.7.0 pure-eval-0.2.2 pycparser-2
.21 pygments-2.16.1 python-json-logger-2.0.7 pytz-2023.3 pyzmq-25.1.0 qtconsole-5.4.3 qtpy-2.3.1 refe
rencing-0.30.2 rfc3339-validator-0.1.4 rfc3986-validator-0.1.1 rpds-py-0.9.2 send2trash-1.8.2 sniffio
-1.3.0 soupsieve-2.4.1 stack-data-0.6.2 terminado-0.17.1 tinycss2-1.2.1 tomli-2.0.1 tornado-6.3.2 tra
itlets-5.9.0 typing-extensions-4.7.1 uri-template-1.3.0 wcwidth-0.2.6 webcolors-1.13 webencodings-0.5
.1 websocket-client-1.6.1 widgetsnbextension-4.0.8
Collecting notebook==6.4.0
```

```
Installing collected packages: notebook
  Attempting uninstall: notebook
    Found existing installation: notebook 7.0.2
    Uninstalling notebook-7.0.2:
      Successfully uninstalled notebook-7.0.2
Successfully installed notebook-6.4.0
(unet) daniel@nacho:/media/daniel/T7/unet-nuclei$
```

Run nuclear segmentation using the corresponding jupyter notebook

Time: 5 hours

Size: 100 GB

- Start the jupyter server via

```
conda activate unet
jupyter notebook
```

```
(base) daniel@nacho:/media/daniel/T7/unet-nuclei$ conda activate unet
(unet) daniel@nacho:/media/daniel/T7/unet-nuclei$ jupyter notebook
[W 15:39:51.335 NotebookApp] Error loading server extension jupyter_lsp
    Traceback (most recent call last):
      File "/usr/share/miniconda3/envs/unet/lib/python3.8/site-packages/notebook/notebookapp.py", lin
e 2030, in init_server_extensions
        func(self)
      File "/usr/share/miniconda3/envs/unet/lib/python3.8/site-packages/jupyter_lsp/serverextension.p
y", line 76, in load_jupyter_server_extension
        nbapp.io_loop.call_later(0, initialize, nbapp, virtual_documents_uri)
    AttributeError: 'NotebookApp' object has no attribute 'io_loop'
[I 2023-08-07 15:39:51.461 LabApp] JupyterLab extension loaded from /usr/share/miniconda3/envs/unet/l
ib/python3.8/site-packages/jupyterlab
[I 2023-08-07 15:39:51.461 LabApp] JupyterLab application directory is /usr/share/miniconda3/envs/une
t/share/jupyter/lab
[I 2023-08-07 15:39:51.461 LabApp] Extension Manager is 'pypi'.
[I 15:39:51.465 NotebookApp] Serving notebooks from local directory: /media/daniel/T7/unet-nuclei
[I 15:39:51.465 NotebookApp] Jupyter Notebook 6.4.0 is running at:
[I 15:39:51.465 NotebookApp] http://localhost:8888/?token=2e82ed99f08b897d1ad92061feb16802def7f25646c
6da90
[I 15:39:51.465 NotebookApp]  or http://127.0.0.1:8888/?token=2e82ed99f08b897d1ad92061feb16802def7f25
646c6da90
[I 15:39:51.465 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to sk
ip confirmation).
[C 15:39:51.486 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/daniel/.local/share/jupyter/runtime/nbserver-32428-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=2e82ed99f08b897d1ad92061feb16802def7f25646c6da90
     or http://127.0.0.1:8888/?token=2e82ed99f08b897d1ad92061feb16802def7f25646c6da90
/usr/share/miniconda3/envs/unet/lib/python3.8/json/encoder.py:257: UserWarning: date_default is depre
cated since jupyter_client 7.0.0. Use jupyter_client.jsonutil.json_default.
  return _iterencode(o, 0)
```

- Open and run the jupyter notebook located in `unet/notebooks/rohban_segmentation.ipynb`

Out[7]:
```
['/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744',
 '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41749',
 '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41755',
 '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41754',
 '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41756',
 '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41757']
```

In [8]:
```
input_file_list
```

Out[8]:
```
[['/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_k21_s7_w10efe595e-e9f1-4dfa-a301-4d8a19b91d40_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_i13_s4_w13be2d93d-4e04-4527-ba7c-4864ca27cfb1_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_j16_s9_w1b03efd40-1f4c-44bd-8839-64b2986ee5dc_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_m07_s5_w1226d78cd-7c89-45ef-a576-6b062c73477b_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_i04_s5_w1f731236e-632e-4345-bc3a-f449abbfdaf7_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_l02_s2_w1aad1482e-f8e6-4e47-ad0a-f41cef0662cb_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_j04_s4_w1753108f6-de03-42b8-b28e-38bd888f9113_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_l03_s8_w107271ae7-d4f1-4a19-9522-d737eb85dd6d_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
9_j22_s5_w1ece2d048-25e0-4c7d-8f32-19d3e64e9217_illum_corrected.tif',
  '/media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744/taoe005-u2os-72h-cp-a-au0004485
```

In [*]:
```
for i in range(len(input_dirs)):
    print("Process images in {}... Results will be stored in {}".format(input_dirs[i], output_dirs[i]))
    unet_segmenter.segment_image_dir(input_dirs[i], output_dirs[i], file_list=input_file_list[1])
```

```
Process images in /media/daniel/T7/image2reg/data/resources/images/rohban/illum_corrected/41744... Results will be
stored in /media/daniel/T7/image2reg/data/resources/images/rohban/unet_masks/41744

  0%|                                                                                | 0/3456 [00:00<?, ?it/s]2023-08-07 1
6:30:24.487430: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:176] None of the MLIR Optimization Pass
es are enabled (registered 2)
2023-08-07 16:30:24.487853: I tensorflow/core/platform/profile_utils/cpu_utils.cc:114] CPU Frequency: 2599990000 H
z
2023-08-07 16:30:30.941374: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dy
namic library libcudnn.so.8
2023-08-07 16:30:31.575085: I tensorflow/stream_executor/cuda/cuda_dnn.cc:359] Loaded cuDNN version 8600
```

```
(base) daniel@nacho:/media/daniel/T7/image2reg/data/resources/images/rohban/unet
_masks$ ls
41744  41749  41754  41755  41756  41757
(base) daniel@nacho:/media/daniel/T7/image2reg/data/resources/images/rohban/unet
_masks$ ls -l
total 6144
drwxr-xr-x 2 daniel daniel 1048576 Aug  7 17:15 41744
drwxr-xr-x 2 daniel daniel 1048576 Aug  7 18:05 41749
drwxr-xr-x 2 daniel daniel 1048576 Aug  8 10:22 41754
drwxr-xr-x 2 daniel daniel 1048576 Aug  8 07:42 41755
drwxr-xr-x 2 daniel daniel 1048576 Aug  8 11:12 41756
drwxr-xr-x 2 daniel daniel 1048576 Aug  8 11:59 41757
(base) daniel@nacho:/media/daniel/T7/image2reg/data/resources/images/rohban/unet
_masks$
```

Preprocess Rohban imaging data via script

Time: 5 hours

Size: 80 GB

- Run the preprocessing script which will create an output directory in the directory `data/experiments/rohban/images/preprocessing/full_pipeline`

```
conda activate image2reg
python run.py --config config/preprocessing/full_image_pipeline.yml
```

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ python run.py --config config/preprocessing
/full_image_pipeline.yml
Copying filtered images: 100%|          | 17040/17040 [00:20<00:00, 826.89it/s]
100%|                                    | 17040/17040 [2:28:21<00:00,  1.91it/s]
Add aspect ratio cluster information:  14%|| | 2332/17031 [03:34<22:43, 10.78it/Add aspect ratio
cluster information:  14%|| | 2334/17031 [03:35<22:43, 10.7Add aspect ratio cluster information:
 14%|| | 2336/17031 [03:35<22:52, 10.71iAdd aspect ratio cluster information:  14%|| | 2338/17031
[03:35<22:46, 10.75iAdd Add aspect ratio cluster information:  14%|| | 2342/17031 [03:35<22:46, 1
0.75itAdd aspect ratio cluster information:  14%|| | 2344/17031 [03:36<22:39, 10.80itAdd aspect r
atio clusterAdd asAdd asAdd asAdd asAdd aspect rAdd aspect ratio cAdd asAdd aspect ratio cAdd as
Add aspect ratio cluster information: 100%|| | 17031/17031 [25:59<00:00, 10.92it/1, 10.79it/s86it
Save padded images: 100%|                  | 1064795/1064795 [16:53<00:00, 1050.40it/s]
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$
```

- By default all output directories created as a result of running the run.py will be named after the time point when the script was started. For the consecutive analyses please copy the content output directory created by the above script to "`full_pipeline`" directory as shown in the screenshot below

```
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ tree -L 1 data/experiments
/rohban/images/preprocessing/full_pipeline/
data/experiments/rohban/images/preprocessing/full_pipeline/
├── filtered
├── filtered_image_metadata.csv
├── full_image_pipeline.yml
├── logs20230809_081652.log
├── nuclei_images
├── nuclei_ncmo_features.csv.gz
├── padded_nuclei
├── padded_nuclei_metadata.csv.gz
├── processed_image_metadata.csv.gz
└── processed_nuclei_metadata.csv.gz

3 directories, 7 files
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$
```

GEX data

Preprocess scRNA-seq data data via notebook
Time: 5 minutes
Size: < 1 GB

- Start jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the jupyter notebook `notebooks/rohban/ppi/gex_analyses/scgex_preprocessing.ipynb`
- Run all cells in the notebook
- The final cell generates a file that contains the preprocessed gene expression data namely `image2reg/data/experiments/rohban/gex/scrnaseq/fucci_adata.h5` of the preprocessed scRNA-seq data

Preprocess CMAP gene signatures via notebook
Time: 2 minutes
Size: < 1 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the jupyter notebook `notebooks/rohban/ppi/gex_analyses/cmap_preprocessing.ipynb`

- Run all cell in the notebook
- The final cell generates a file that contains the processed CMap gene signatures namely `image2reg/data/experiments/rohban/gex/cmap/mean_l5_signatures_tmp.csv`

Identify impactful OE conditions

Generate required data split files

Time: 5 minutes

Size: 4 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
Jupyter notebook
```

- Start the notebook `notebooks/rohban/other/cv_screen_data_split.ipynb`
- Run all cells in the notebook
- This generate the four-fold stratified group cross-validation data splits required for the screen as shown in the screenshot below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/imag
es/preprocessing/screen_splits$ ls
ACVR1B    BRCA1     DIABLO    HIF1AN      MAP3K8    PER1      RELB      TGFB1
ADAM17    BTRC      DKK1      HRAS        MAP3K9    PHLPP1    RHEB      TGFBR1
AKT1      CARD11    DLL1      HSP90AA1    MAPK1     PIK3CA    RHOA      TGFBR2
AKT1S1    CASP8     DUSP1     HSP90B1     MAPK13    PIK3CB    RICTOR    TNFAIP3
AKT2      CASP9     DVL1      HSPA5       MAPK14    PIK3CD    RIPK1     TP53
AKT3      CCND1     DVL2      IKBKB       MAPK3     PIK3R1    RPS6KB1   TRAF2
APAF1     CCNE1     DVL3      IKBKE       MAPK7     PIK3R2    RPTOR     TRAF3
APC       CDC42     E2F1      IRAK1       MAPK8     PKIA      SDHA      TRAF5
ARAF      CDK2      eGFP      IRAK4       MAPK9     PPARGC1A  SGK3      TRAF6
ARNTL     CDK4      EGLN1     IRGM        MAPKAP1   PPP2R5C   SLIRP     TSC1
ATF2      CDKN1A    EIF2A     IRS1        MCL1      PRKAA1    SMAD3     TSC2
ATF4      CEBPA     EIF4E     JAG1        MEK1      PRKACA    SMAD4     VEGFC
ATF6      CHUK      EIF4EBP1  JAK2        MKNK1     PRKACB    SMAD5     VHL
ATG16L1   CLOCK     ELK1      JUN         MLST8     PRKACG    SMAD7     WNT5A
ATG5      CREB1     ERBB2     KRAS        MOS       PRKCA     SMO       WWTR1
ATM       CREBBP    ERG       LacZ        MYD88     PRKCE     SMURF2    XBP1
AXIN2     CRY1      ERN1      LRPPRC      NFKB1     PRKCZ     SOCS3     XIAP
BAMBI     CSNK1A1   FGFR3     Luciferase  NFKB2     PSENEN    SRC       YAP1
BAX       CSNK1E    FH        MAP2K1      NFKBIA    PTEN      SREBF1
BCL2L1    CTNNB1    FOXO1     MAP2K3      NFKBIB    RAC1      STAT1
BCL2L11   CXXC4     FURIN     MAP2K4      NFKBIE    RAF1      STAT3
BECN1     CYLD      GLI1      MAP3K11     NOTCH1    RB1       STK11
BMP2      DDIT3     GRB10     MAP3K2      NOTCH2    RBPJ      STK3
BMPR1B    DDIT4     GSK3B     MAP3K5      PAK1      REL       TBK1
BRAF      DEPTOR    HIF1A     MAP3K7      PDPK1     RELA      TCF4
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/imag
es/preprocessing/screen_splits$
```

Run specificity screen
Time: 200 - 800 hours[1]
Size: 150 GB

- Run the specificity screen via

```
conda activate image2reg
bash run_screen.sh
```

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ bash scripts/experiments/run_screen.sh
Epoch 0 progress for train phase:   7%|   | 22/314 [00:55<06:15,  1.28s/it]
```

Rename outputs
Time: 1 minute
Size: n/a

- Rename the directories created during running the specificity screen such that each fold directory contains one directory named after one of the tested 190 OE conditions
- To rename the directories run

```
conda activate image2reg
python scripts/experiments/rename_screen_dirs -root_dir
data/experiments/rohban/images/screen/nuclei_region
```

Analyze results
Time: 1 hour
Size: > 1GB

- Start the jupyter server in the conda environment

```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/rohban/image/screen/screen_analyses_cv_final.ipynb`
- Run all cells
- This creates a summary of the screen results and saves it as `data/experiments/rohban/images/screen/specificity_screen_results_cv.csv` as shown in the screenshot below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ cd data/experiments/rohban/
images/screen/
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/imag
es/screen$ ls
nuclei_region  specificity_screen_results_cv.csv
```

[1] Depending on the speed of the I/O of the disk where the images are stored. Using a CPU and not a GPU will increase the time by an order of magnitude.

Gene perturbation embeddings

General setup using all OE conditions

Generate data splits
Time: 2 minutes
Size: > 1 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/rohban/other/cv_specific_targets_data_split.ipynb`
- Run all cells
- This creates the required metadata csv-files for the individual splits of the stratified four-fold group cross-validation in `data/experiments/images/preprocessing/specific_targets_cv_stratified` as seen in the screenshot below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/imag
es/preprocessing/specific_targets_cv_stratified$ ls
nuclei_md_test_fold_0.csv  nuclei_md_train_fold_0.csv  nuclei_md_val_fold_0.csv
nuclei_md_test_fold_1.csv  nuclei_md_train_fold_1.csv  nuclei_md_val_fold_1.csv
nuclei_md_test_fold_2.csv  nuclei_md_train_fold_2.csv  nuclei_md_val_fold_2.csv
nuclei_md_test_fold_3.csv  nuclei_md_train_fold_3.csv  nuclei_md_val_fold_3.csv
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/imag
es/preprocessing/specific_targets_cv_stratified$
```

Run four-fold 41 OE + 1 control classification
Time: 20 - 30 hours[2]
Size: 4 GB

- Run the bash script performing the four-fold stratified grouped cross-validation approach via

```
conda activate image2reg
bash scripts/experiments/run_selected_targets.sh
```

- This will perform all four folds and the output after starting the script should look like the screenshot below

[2] Depending on the speed of the I/O of the disk where the images are stored. Using a CPU and not a GPU will increase the time by an order of magnitude.

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ bash scripts/experiments/ru
n_selected_targets.sh
/media/paysan_d/T7/image2reg/src/data/datasets.py:148: DtypeWarning: Columns (11
) have mixed types. Specify dtype option on import or set low_memory=False.
  self.metadata = pd.read_csv(self.metadata_file, index_col=0)
Epoch 0 progress for train phase:   2%|▏       | 62/2903 [00:19<41:33,  1.14it/s]
```

Restructure outputs

- The previously run script will create output directories named after the time step when the experiment was conducted in `data/experiments/rohban/images/embeddings/four_fold_cv/fold{0,1,2,3}`
- Simply copy all contents of these fold-directories to their parent directory, i.e. such that all files in `fold0/<timestamp>` are position in `fold0`
- Remove the timestamp directories
- The expected file structure should look as shown below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/i
mages/embeddings/four_fold_cv$ tree .
.
├── fold_0
│   ├── best_model_weights.pth
│   ├── confusion_matrix_test.png
│   ├── confusion_matrix_train.png
│   ├── confusion_matrix_val.png
│   ├── epoch_0
│   │   └── classifier.pth
│   ├── fold_0.yml
│   ├── logs20220317_100439.log
│   ├── plotted_fitting_hist.png
│   ├── test
│   │   └── classifier.pth
│   ├── test_cmatrix.csv
│   ├── test_latents.h5
│   ├── train_cmatrix.csv
│   ├── train_latents.h5
│   ├── val_cmatrix.csv
│   └── val_latents.h5
├── fold_1
│   ├── best_model_weights.pth
│   ├── confusion_matrix_test.png
│   ├── confusion_matrix_train.png
│   ├── confusion_matrix_val.png
│   ├── epoch_0
│   │   └── classifier.pth
│   ├── fold_1.yml
│   ├── logs20220317_132043.log
│   ├── plotted_fitting_hist.png
│   ├── test
│   │   └── classifier.pth
│   ├── test_cmatrix.csv
│   ├── test_latents.h5
│   ├── train_cmatrix.csv
│   ├── train_latents.h5
│   ├── val_cmatrix.csv
│   └── val_latents.h5
├── fold_2
```

Analyze results

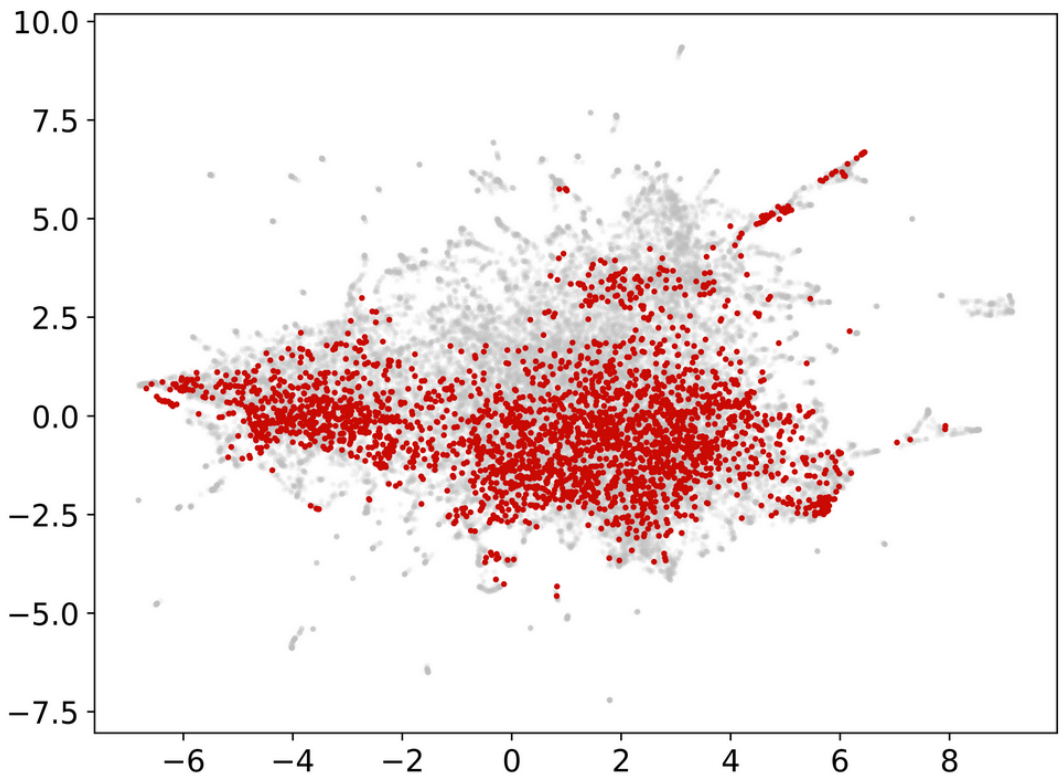
Time: 30 minutes
Size: n/a

- Start the jupyter server in the conda environment

```
conda activate image2reg
jupyter notebook
```

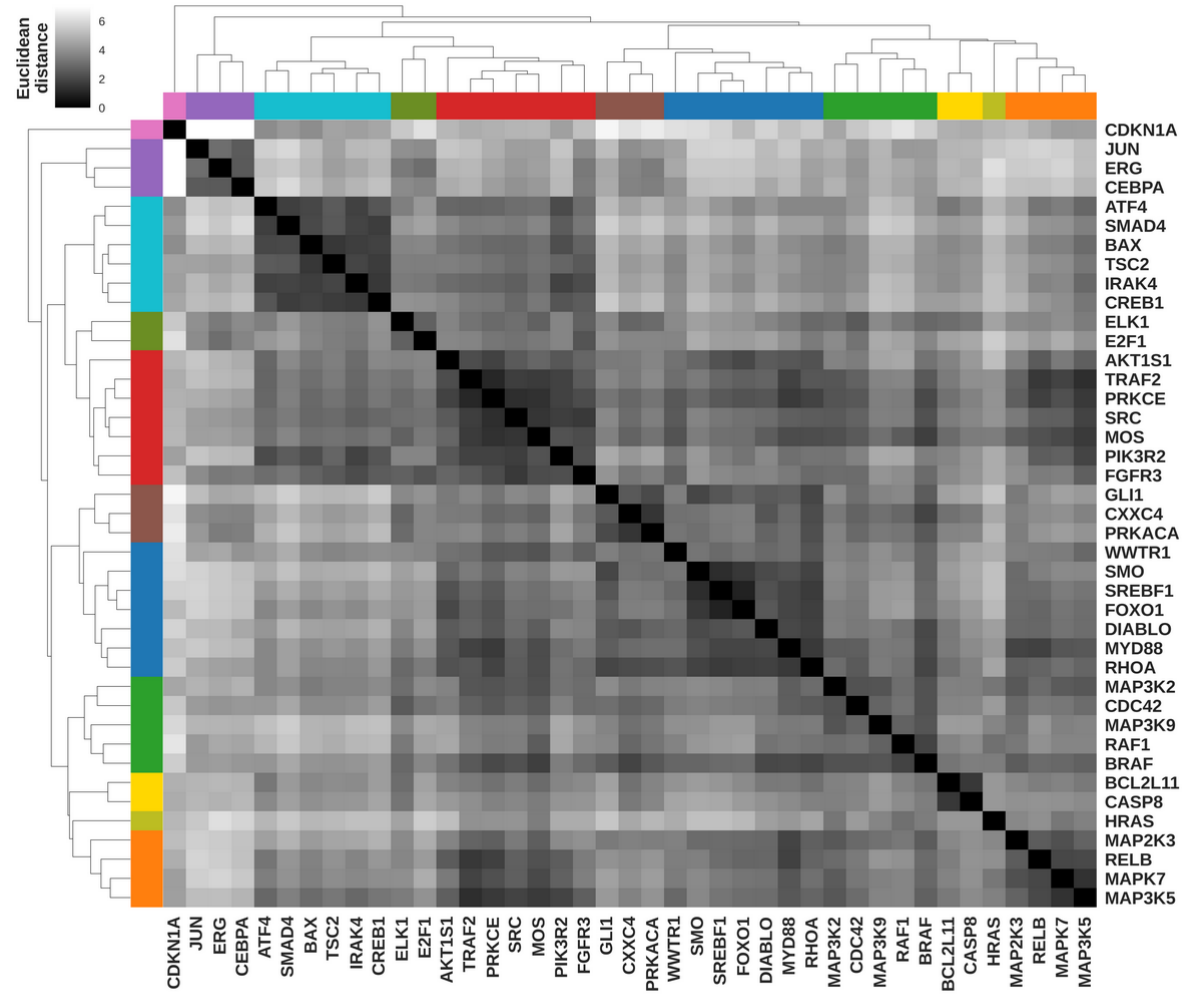
- Start the notebook `notebooks/rohban/image/embedding/image_embeddings_analysis.ipynb`
- Run all cells
- This produces the e.g. the Fig. 2C of the manuscript as seen e.g in the screenshot below

```
#         prop=dict(size=18),
#     )
#     #     for lh in ax.get_legend().legendHandles:
#     #         lh.set_alpha(1)
#     #         lh._sizes = [140]
# ax.get_legend().set_title("Condition", prop={"size": "20"})
#     ax.get_legend().set_title("")
# ax.set_xlabel("umap_0", size=18)
# ax.set_ylabel("umap_1", size=18)
ax.set_xlabel("")
ax.set_ylabel("")
plt.xticks(size=14)
plt.yticks(size=14)
plt.show()
plt.close()
```



- Start the notebook `notebooks/rohban/image/embedding/gene_perturbation_cluster_analysis.ipynb`
- Run all cells to e.g. reproduce the Fig. 2e of the manuscript as shown in the screenshot below

```
  index=mean_physical_space_embs.loc[oe_targets].index,
<ipython-input-13-888b765b4761>:4: FutureWarning: Passing a set as an indexer is deprecated and will raise in a futu
re version. Use a list instead.
  columns=mean_physical_space_embs.loc[oe_targets].index,
```




```
print(list(mean_phy_space_embs_dist))
```

Leave-one-target-out cross validation setup

Generate data splits

Time: 10 minutes

Size: 3 GB

- Start the jupyter server in the conda environment

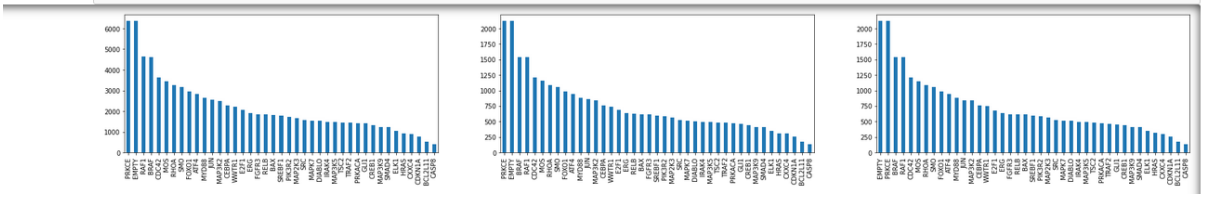
```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/rohban/other/loto_data_splits.ipynb`
- Run all cells like e.g. the ones shown in the screenshot below

In [13]:
```
target_list = sorted(list(spec_orf_targets) + ["EMPTY"])
print(target_list)
loto_data = get_loto_data_splits(
    data=md,
    label_col=label_col,
    target_list=target_list,
    group_col=group_col,
    random_state=random_state,
)
```

```
['AKT1S1', 'ATF4', 'BAX', 'BCL2L11', 'BRAF', 'CASP8', 'CDC42', 'CDKN1A', 'CEBPA', 'CREB1', 'CXXC4', 'DIABLO', 'E2F1
', 'ELK1', 'EMPTY', 'ERG', 'FGFR3', 'FOXO1', 'GLI1', 'HRAS', 'IRAK4', 'JUN', 'MAP2K3', 'MAP3K2', 'MAP3K5', 'MAP3K9',
'MAPK7', 'MOS', 'MYD88', 'PIK3R2', 'PRKACA', 'PRKCE', 'RAF1', 'RELB', 'RHOA', 'SMAD4', 'SMO', 'SRC', 'SREBF1', 'TRAF
2', 'TSC2', 'WWTR1']
100%|██████████| 42/42 [02:19<00:00,  3.31s/it]
```

In [14]:
```
for i in range(len(target_list)):
    fig, ax = plt.subplots(figsize=[30, 4], ncols=3)
    ax = ax.flatten()
    loto_data["train"][i].gene_symbol.value_counts().plot(
        kind="bar", figsize=(30, 4), ax=ax[0]
    )
    loto_data["val"][i].gene_symbol.value_counts().plot(
        kind="bar", figsize=(30, 4), ax=ax[1]
    )
    loto_data["test"][i].gene_symbol.value_counts().plot(
        kind="bar", figsize=(30, 4), ax=ax[2]
    )
    plt.show()
    plt.close()
```



- This produces a number of csv files that describe the data splits for the four-fold stratified grouped cross-validation for the leave-one-target-out inference stored in `data/experiments/rohban/images/preprocessing/loto_cv_stratified` and results in a file structure as shown in the screenshot below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/images/preprocessing$ ls
full_pipeline       screen_splits
loto_cv_stratified  specific_targets_cv_stratified
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/images/preprocessing$ tree loto_cv_stratified/
loto_cv_stratified/
├── AKT1S1
│   ├── nuclei_md_AKT1S1.csv
│   ├── nuclei_md_AKT1S1_loo_test.csv
│   ├── nuclei_md_AKT1S1_loo_train.csv
│   └── nuclei_md_AKT1S1_loo_val.csv
├── ATF4
│   ├── nuclei_md_ATF4.csv
│   ├── nuclei_md_ATF4_loo_test.csv
│   ├── nuclei_md_ATF4_loo_train.csv
│   └── nuclei_md_ATF4_loo_val.csv
├── BAX
│   ├── nuclei_md_BAX.csv
│   ├── nuclei_md_BAX_loo_test.csv
│   ├── nuclei_md_BAX_loo_train.csv
│   └── nuclei_md_BAX_loo_val.csv
├── BCL2L11
│   ├── nuclei_md_BCL2L11.csv
│   ├── nuclei_md_BCL2L11_loo_test.csv
│   ├── nuclei_md_BCL2L11_loo_train.csv
│   └── nuclei_md_BCL2L11_loo_val.csv
├── BRAF
│   ├── nuclei_md_BRAF.csv
│   ├── nuclei_md_BRAF_loo_test.csv
│   ├── nuclei_md_BRAF_loo_train.csv
│   └── nuclei_md_BRAF_loo_val.csv
├── CASP8
│   ├── nuclei_md_CASP8.csv
│   ├── nuclei_md_CASP8_loo_test.csv
```

Run 41 multi-class classification

Time: 40 - 100 hours[3]

Size: 40 GB

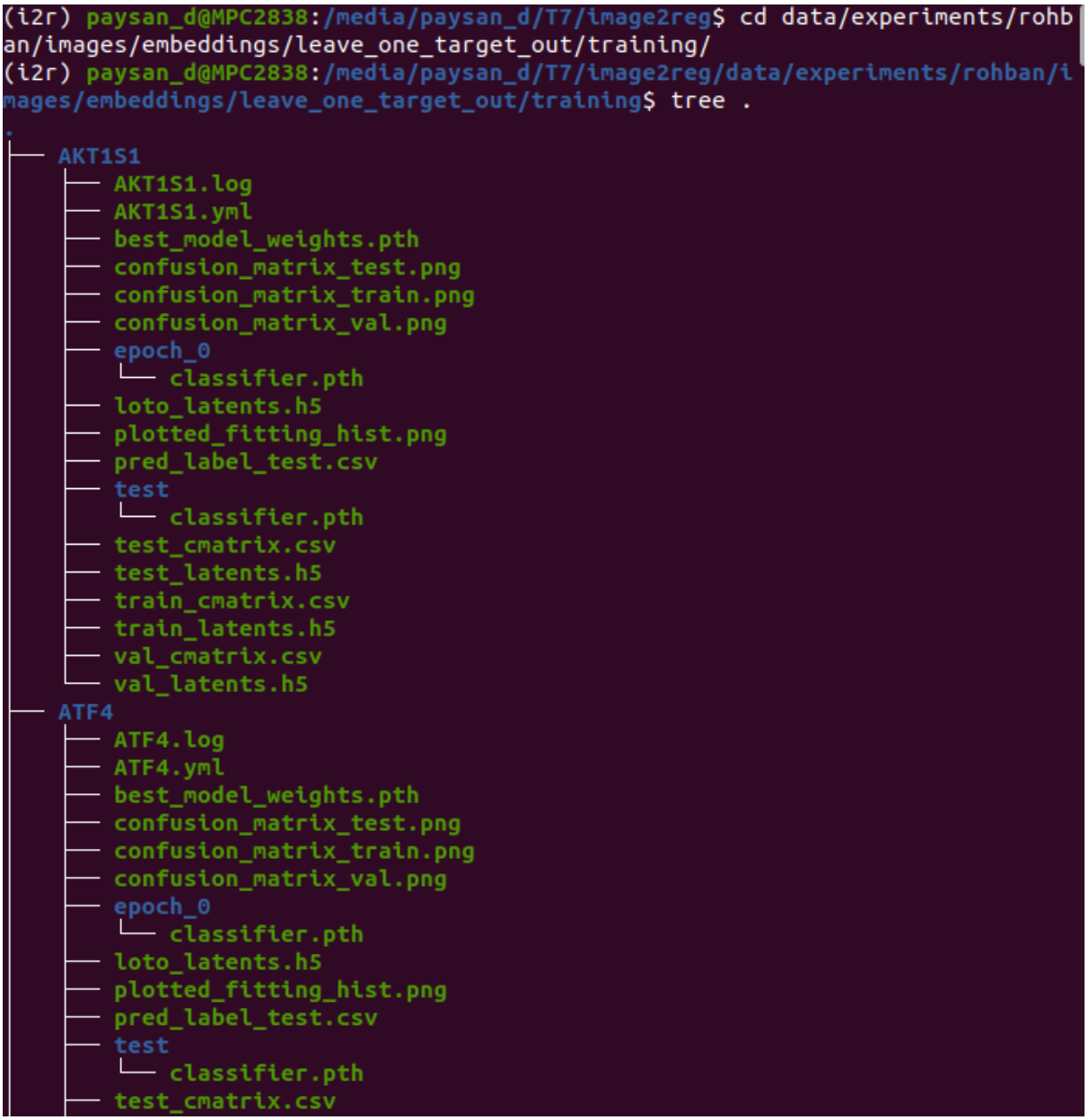
- Start the leave-one-target-out classification experiment via

```
conda activate image2reg
bash scripts/experiments/run_loto_selected_targets.sh
```

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ bash scripts/experiments/run_loto_selected_targets.sh
Epoch 0 progress for train phase:   2%|     | 30/1403 [01:45<10:58,  2.08it/s]
```

[3] Depending on the speed of the I/O of the disk where the images are stored. Using a CPU and not a GPU will increase the time by an order of magnitude.

- The results will be stored in `data/experiments/rohban/images/embeddings/leave_one_target_out/training`
- Place all contents in the timestamp directories located in `training/<target>` directly into the `training/<target>` directory to recreate a file structure like the one shown in the screenshot below



Analyze results

Time: 40 minutes

Size: 1 GB

- Start jupyter server in conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/rohban/image/embedding/image_embeddings_analysis_loto.ipynb`
- Run all cells

1.5 4. Save data

We finally save the inferred cluster solutions as well as the the image embeddings for the training and test set as those will be later used to infer the regulatory space and/or the translational mapping.

[34]:
```
output_dir = (
    "../../../../data/experiments/rohban/images/embeddings/leave_one_target_out/embeddings"
)
targets = sorted(list(train_latents_dict.keys()))
for target in tqdm(targets):
    target_output_dir = os.path.join(output_dir, target)
    os.makedirs(target_output_dir, exist_ok=True)
    train_latents = train_latents_dict[target]
    train_latents.to_hdf(
        os.path.join(target_output_dir, "train_latents.h5"), key="data"
    )
    test_latents = test_latents_dict[target]
    test_latents.to_hdf(os.path.join(target_output_dir, "test_latents.h5"), key="data")
    loto_latents = loto_latents_dict[target]
    phy_embs = pd.concat([train_latents, loto_latents])
    phy_embs.to_hdf(os.path.join(target_output_dir, "phy_embs.h5"), key="data")
    train_mean_clusters = mean_clusters_dict[target]
    train_mean_clusters.to_csv(
        os.path.join(target_output_dir, "train_phy_mean_clusters.csv")
    )
```

100% 42/42 [02:47<00:00, 3.18s/it]

- This will create a number of files located in the output directory `data/experiments/rohban/images/embeddings/leave_one_target_out_embeddings` and a file structure as shown in the screenshot below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/images/embeddings/leave_one_target_out$ ls
embeddings  training
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/images/embeddings/leave_one_target_out$ cd embeddings/
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/images/embeddings/leave_one_target_out/embeddings$ tree
.
├── AKT1S1
│   ├── phy_embs.h5
│   ├── phy_mean_full_clusters.csv
│   ├── test_latents.h5
│   ├── train_latents.h5
│   └── train_phy_mean_clusters.csv
├── ATF4
│   ├── phy_embs.h5
│   ├── phy_mean_full_clusters.csv
│   ├── test_latents.h5
│   ├── train_latents.h5
│   └── train_phy_mean_clusters.csv
├── BAX
│   ├── phy_embs.h5
│   ├── phy_mean_full_clusters.csv
```

Identify cell-type specific gene-gene interactome

Prepare human PPI for Steiner tree analysis

Time: 5 minutes

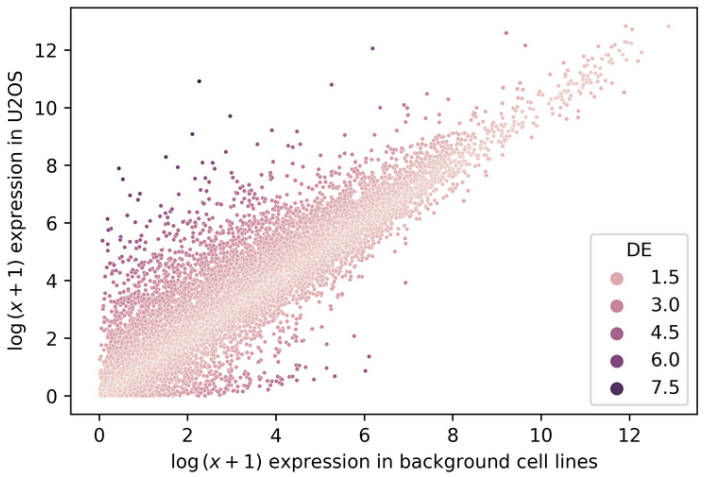
Size: > 1 GB

- Start the jupyter server in the conda environment

```
conda activate image2reg
jupyter notebook
```

- Run the notebook `notebooks/rohban/ppi/preprocesssing/inference_preparation_full_pruning.ipynb` and run all cells
- This will preprocess all the data and save the preprocessed protein-protein interactome as a pickle file in `data/experiments/rohban/interactome/preprocessing` and also produce components of the e.g. the Fig. 3a as shown below

```
In [55]: fig, ax = plt.subplots(figsize=[6, 4])
ax = sns.scatterplot(
    data=marker_results,
    x="avg_log_expr_other",
    y="log_expr_target",
    s=5,
    hue="abs_log_fc",
    #     palette={
    #         "up-regulated": "forestgreen",
    #         "not DEGs": "black",
    #         "down-regulated": "tab:red",
    #     },
    #     hue_order=["up-regulated", "not DEGs", "down-regulated"],
    cmap="green",
)
ax.legend(title="DE")
ax.set_ylabel(r"$\log{(x+1)}$ expression in U2OS")
ax.set_xlabel(r"$\log{(x+1)}$ expression in background cell lines")
plt.show()
```



Identify U2OS-specific GGI via the PCST algorithm

Time: 10 minutes

Size: > 1 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the jupyter notebook `notebooks/rohban/ppi/inference/interactome_inference_final.ipynb` and run all cells which produces e.g the output below

[31]:
```
augmented_pcst_spearman_results = analyze_pcst_sensitivity_analyses_results(
    augmented_pcst_spearman_dict,
    target_nodes=orf_targets,
    spec_targets=specific_targets,
)
augmented_pcst_spearman_results.loc[
    augmented_pcst_spearman_results["n_spec_target_nodes"]
    == np.max(augmented_pcst_spearman_results["n_spec_target_nodes"])
].sort_values("beta")
```

```
Analyze tree:   0%|          | 0/99 [00:00<?, ?it/s]/home/paysan_d/miniconda3/envs/i2r/lib/p
ython3.8/site-packages/numpy/core/fromnumeric.py:3474: RuntimeWarning: Mean of empty slice.
  return _methods._mean(a, axis=axis, dtype=dtype,
/home/paysan_d/miniconda3/envs/i2r/lib/python3.8/site-packages/numpy/core/_methods.py:189: R
untimeWarning: invalid value encountered in double_scalars
  ret = ret.dtype.type(ret / rcount)
/home/paysan_d/miniconda3/envs/i2r/lib/python3.8/site-packages/numpy/core/_methods.py:264: R
untimeWarning: Degrees of freedom <= 0 for slice
  ret = _var(a, axis=axis, dtype=dtype, out=out, ddof=ddof,
/home/paysan_d/miniconda3/envs/i2r/lib/python3.8/site-packages/numpy/core/_methods.py:222: R
untimeWarning: invalid value encountered in true_divide
  arrmean = um.true_divide(arrmean, div, out=arrmean, casting='unsafe',
/home/paysan_d/miniconda3/envs/i2r/lib/python3.8/site-packages/numpy/core/_methods.py:256: R
untimeWarning: invalid value encountered in double_scalars
  ret = ret.dtype.type(ret / rcount)
Analyze tree: 100%|██████████| 99/99 [00:22<00:00,  4.45it/s]
```

t[31]:

	beta	n_nodes	n_edges	n_con
augmented_ppi_confidence_0594_hub_999_pruned_ccle_abslogfc_orf_maxp_spearmanr_cv_b_2.6	2.6	249	526	

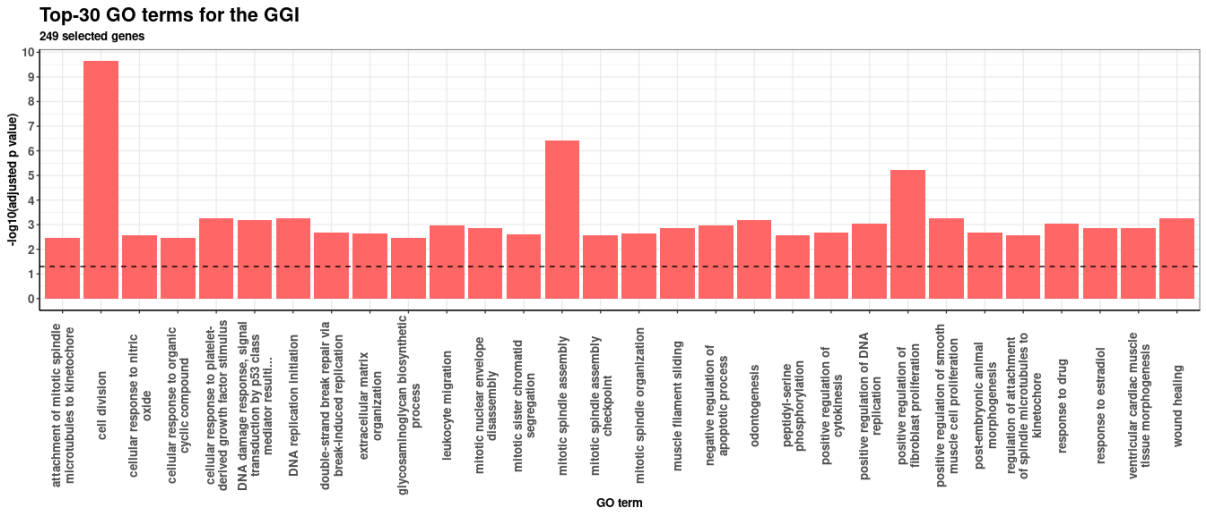
- The cells also save the inferred gene-gene interactome as a pickle and .graphml file in `data/experiments/rohban/interactome/inference_results`
- To visualize the inferred network and reproduce the visualization of the gene-gene interactome in Fig. 3a please open the .graphml file in Cytoscape.

Analyze the results

Time: 5 minutes

Size: > 1GB

- Start RStudio and open the Rmd Notebook `notebooks/rohban/ppi/other/go_analysis_pcst_solution.Rmd`
- Set the working directory in the first cell to the location of the directory `image2reg`
- Run all cells to reproduce the GO results for the PCST solution shown in Fig. S12


Top-30 GO terms for the GGI
249 selected genes
-log10(adjusted p value)
10
9
8
7
6
5
4
3
2
1
0
attachment of mitotic spindle microtubules to kinetochore
cell division
cellular response to nitric oxide
cellular response to organic cyclic compound
cellular response to platelet-derived growth factor stimulus
DNA damage response, signal transduction by p53 class mediator resulti...
DNA replication initiation
double-strand break repair via break-induced replication
extracellular matrix organization
glycosaminoglycan biosynthetic process
leukocyte migration
mitotic nuclear envelope disassembly
mitotic sister chromatid segregation
mitotic spindle assembly
mitotic spindle assembly checkpoint
mitotic spindle organization
muscle filament sliding
negative regulation of apoptotic process
odontogenesis
peptidyl-serine phosphorylation
positive regulation of cytokinesis
positive regulation of DNA replication
positive regulation of fibroblast proliferation
positive regulation of smooth muscle cell proliferation
post-embryonic animal morphogenesis
regulation of attachment of spindle microtubules to kinetochore
response to drug
response to estradiol
ventricular cardiac muscle tissue morphogenesis
wound healing
GO term

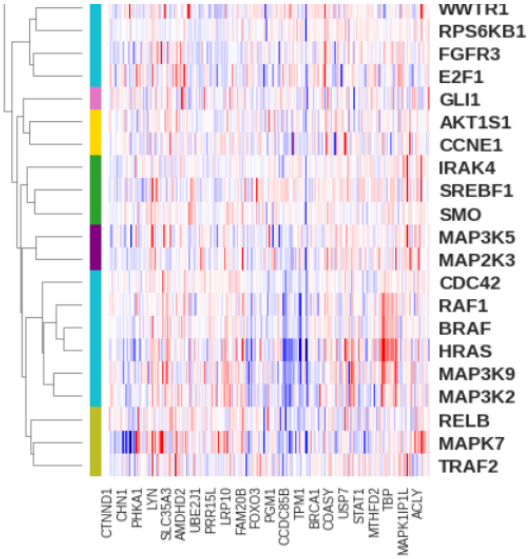
Regulatory gene embeddings

Compute leave-one-target out gene embeddings
Time: 10 hours
Size: 25 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/rohban/ppi/gex_analyses/cmap_full_clustering.ipynb`
- Run all cells to generate the file `data/experiments/rohban/other/mean_cmap_sig_clusters_all_covered_nodes.csv` as seen in the screenshot below



4. Data export

Finally, we export the CMap gene signatures for the selected nodes as well as the identified clustering structure.

```
In [13]: mean_l5_clusters.to_csv(
             "../../../../data/experiments/rohban/other/mean_cmap_sig_clusters_all_covered_nodes.csv"
         )
```

- Start the notebook `notebooks/notebooks/rohban/ppi/embeddings/gae_gene_embs.ipynb`
- Run all cells this will for different choices of the hyperparameters weighing the different loss components for the GCAE train the graph autoencoder
- The generated regulatory gene embeddings are saved in `data/experiments/rohban/images/embeddings/leave_one_target_out/embeddings/<condition>/spearman_sol` ,where condition is each of the 41 impactful OE conditions

- This also generates the output of all regulatory embeddings and the tsNE plot shown in Fig. 3b as well as the clustering that is assessed in Fig. 3c of the manuscript.

Analyze results
Time: 10 minutes
Size: >1GB

- Start the jupyter server in the conda environment

```
conda activate image2reg
jupyter notebook
```

- Open the notebook notebooks/rohban/ppi/embeddings/gene_embedding_clustering.ipynb
- Run all cells
- This saves the clustering solution of the inferred regulatory gene embeddings in `data/experiments/rohban/cluster_infos/all_gene_embeddings_clusters.csv`

```
(base) paysan_d@MPC2838:~$ cd /media/paysan_d/T7/image2reg
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ cd data/experiments/rohban
/interactome/cluster_infos/
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/int
eractome/cluster_infos$ ls
all_gene_embeddings_clusters.csv
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/rohban/int
eractome/cluster_infos$
```

- Start RStudio and open the notebook `notebooks/rohban/ppi/embeddings/gene_embedding_cluster_analyses.Rmd`
- Run all chunks to reproduce e.g. Fig. 3c

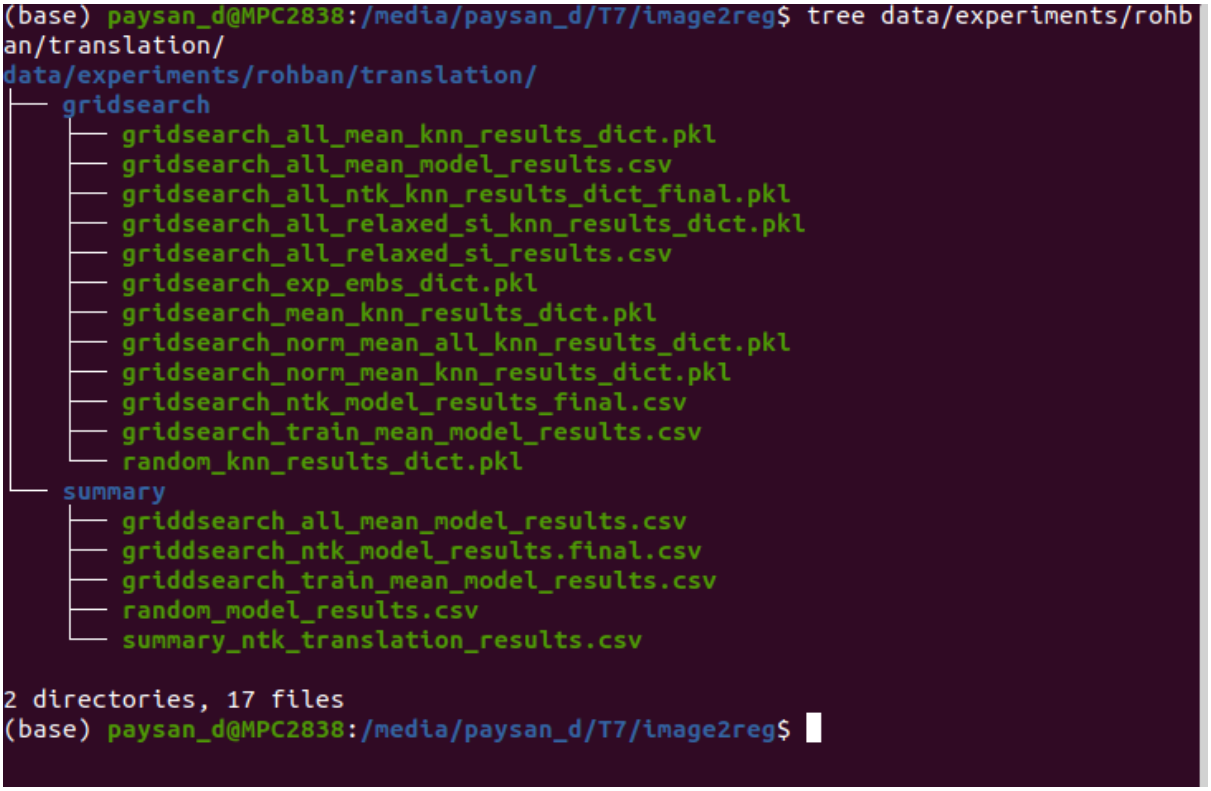
Mapping gene perturbation to regulatory gene embeddings

Run gridsearch and analyze results
Time: 40 hours
Size: 4 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

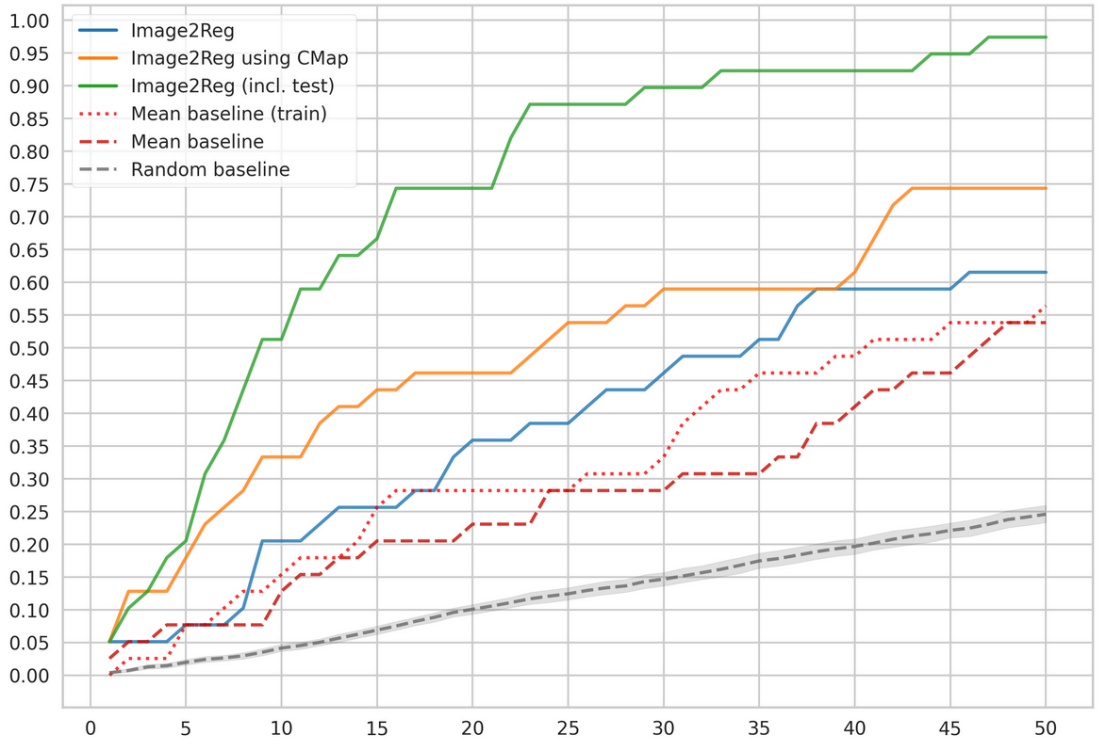
- Start the notebook `notebooks/rohban/translation/mapping/translational_mapping_loto_gridsearch_final.ipynb`
- Run all cells to rerun the gridsearch approach for the NTK regression to map from the gene perturbation to regulatory gene embeddings
- This will create a number of files located at `data/experiments/rohban/translation` as shown in the screenshot below



- Additionally this notebook also will e.g. plot the Fig. 4b of the manuscript as shown in the screenshot below

```
    mean_baseline=mean_baseline_results,
    mean_train_baseline=mean_train_baseline_results,
    figsize=[10, 7],
    cmap="tab10",
    legend_title="",
)
ax.set_xlabel("")
ax.set_ylabel("")
```

Out[89]: Text(0, 0.5, '')



Additional validation using JUMP-CP

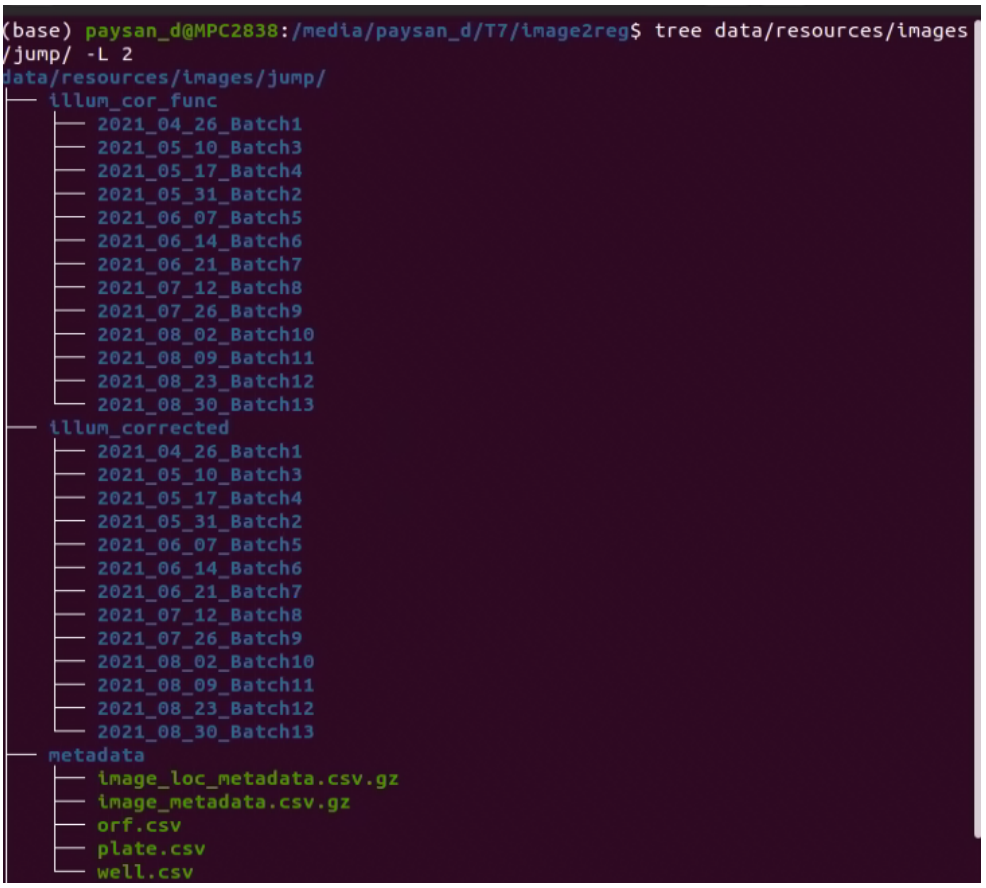
Obtain and prepare data

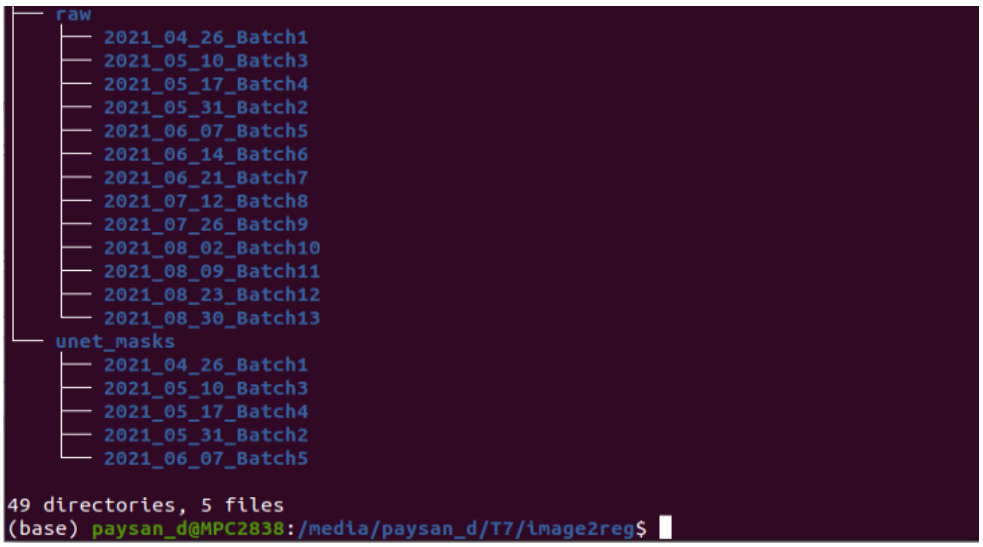
Download JUMP-CP data
Time: 4 hours
Size: 105 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the jupyter notebook `notebooks/jump/eda/data_extraction.ipynb`
- Run all cells to download the image data from the JUMP-CP data set for the selected OE conditions including the illumination corrected images
- All generated data gets downloaded to `data/resources/images/jump` as seen in the screenshot below





Run nuclear segmentation using the corresponding jupyter notebook

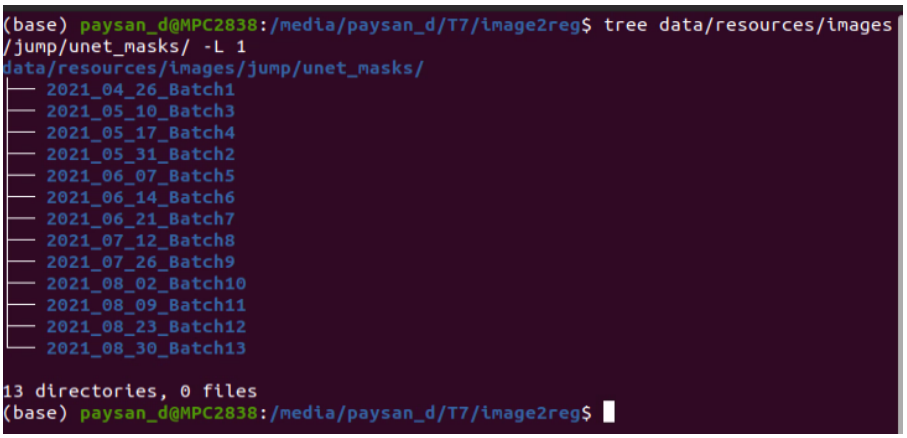
Time: 3 hours

Size: 100 GB

- Start the jupyter server in the unet conda environment via

```
conda activate unet
jupyter notebook
```

- Open and run the jupyter notebook located in `unet/notebook/jump_segmentation.ipynb` (Please be aware that this is not a path in the `image2reg` directory but the `unet-nuclei` directory you have cloned earlier → please refer to the respective section for the Rohban data set in this protocol for more information)
- Running all cells generates the segmentation masks for all images and stores those in image2reg/data/resources/images/jump/unet_masks as shown in the screenshot below



Preprocess Rohban imaging data via script
Time: 100 hours
Size: 300 GB

- Run the preprocessing script which will create an output directory in the directory `data/experiments/jump/images/preprocessing/full_pipeline`

```
conda activate image2reg
python run.py --config config/preprocessing/full_image_pipeline_jump.yml
```

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ python run.py --config config/preprocessing/full_image
_pipeline_jump.yml

data/resources/images/jump/metadata/image_loc_metadata.csv.gz
  0%|                                                    | 100/20742 [01:28<7:12:02,  1.26s/it]
```

- This will run the preprocessing and store the outputs in `data/experiments/jump/images/preprocessing`
- By default all output directories created as a result of running the run.py will be named after the time point when the script was started.
- For the consecutive analyses please copy the content output directory created by the above script to "`full_pipeline`" directory as shown in the screenshot below
-

```
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/images/preprocessing
/full_pipeline$ ls
full_image_pipeline_jump.yml  nuclei_ncmo_features.csv.gz    processed_image_metadata.csv.gz
logs20230303_101047.log       padded_nuclei                  processed_nuclei_metadata.csv.gz
nuclei_images                 padded_nuclei_metadata.csv.gz
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/images/preprocessing
/full_pipeline$
```

Image and gene perturbation embeddings

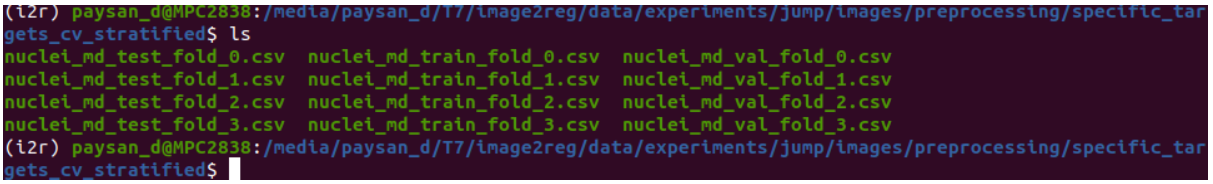
Prepare the 4-fold cross-validated classification
Time: 5 minutes
Size: 1 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/rohban/other/cv_specific_targets_data_split_jump.ipynb`
- Run all cells to create the the metadata files that define the split of the data for the four-fold cross-validation

- The created files are stored in `data/experiments/jump/images/preprocessing/specific_targets_cv_stratified` as shown in the screenshot below



Run four-fold 31+1 classification
Time: 12 hours
Size: 8 GB

- Run the training of the CNN ensemble on the image data from the JUMP data set in the conda environment via

```
conda activate image2reg
python run.py --config
config/image_embedding/specific_targets/cv_jump/nuclei_region/fold_0.yml
python run.py --config
config/image_embedding/specific_targets/cv_jump/nuclei_region/fold_1.yml
python run.py --config
config/image_embedding/specific_targets/cv_jump/nuclei_region/fold_2.yml
python run.py --config
config/image_embedding/specific_targets/cv_jump/nuclei_region/fold_3.yml
```

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ python run.py --config  config/image_embedding/specifi
c_targets/cv_jump/nuclei_region/fold_0.yml
Epoch 0 progress for train phase: 100%|                                        | 2125/2125 [14:06<00:00,  2.51it/s]
Epoch 0 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.19it/s]
Epoch 1 progress for train phase: 100%|                                        | 2125/2125 [13:59<00:00,  2.53it/s]
Epoch 1 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.19it/s]
Epoch 2 progress for train phase: 100%|                                        | 2125/2125 [13:51<00:00,  2.56it/s]
Epoch 2 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.19it/s]
Epoch 3 progress for train phase: 100%|                                        | 2125/2125 [13:51<00:00,  2.56it/s]
Epoch 3 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.18it/s]
Epoch 4 progress for train phase: 100%|                                        | 2125/2125 [13:52<00:00,  2.55it/s]
Epoch 4 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.21it/s]
Epoch 5 progress for train phase: 100%|                                        | 2125/2125 [13:49<00:00,  2.56it/s]
Epoch 5 progress for val phase: 100%|                                          | 527/527 [01:42<00:00,  5.16it/s]
Epoch 6 progress for train phase: 100%|                                        | 2125/2125 [13:50<00:00,  2.56it/s]
Epoch 6 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.20it/s]
Epoch 7 progress for train phase: 100%|                                        | 2125/2125 [13:52<00:00,  2.55it/s]
Epoch 7 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.20it/s]
Epoch 8 progress for train phase: 100%|                                        | 2125/2125 [13:50<00:00,  2.56it/s]
Epoch 8 progress for val phase: 100%|                                          | 527/527 [01:40<00:00,  5.23it/s]
Epoch 9 progress for train phase: 100%|                                        | 2125/2125 [13:50<00:00,  2.56it/s]
Epoch 9 progress for val phase: 100%|                                          | 527/527 [01:41<00:00,  5.19it/s]
Epoch 10 progress for train phase: 100%|                                       | 2125/2125 [13:50<00:00,  2.56it/s]
Epoch 10 progress for val phase: 100%|                                         | 527/527 [01:41<00:00,  5.20it/s]
Epoch 11 progress for train phase: 100%|                                       | 2125/2125 [13:50<00:00,  2.56it/s]
Epoch 11 progress for val phase: 100%|                                         | 527/527 [01:40<00:00,  5.23it/s]
Epoch 12 progress for train phase: 100%|                                       | 2125/2125 [13:51<00:00,  2.56it/s]
Epoch 12 progress for val phase: 100%|                                         | 527/527 [01:42<00:00,  5.16it/s]
Epoch -1 progress for test phase: 100%|                                        | 884/884 [02:48<00:00,  5.24it/s]
Compute predictions: 100%|                                                     | 2125/2125 [06:41<00:00,  5.29it/s]
Balanced accuracy 0.8597111223126483
Compute predictions: 100%|                                                     | 527/527 [01:41<00:00,  5.21it/s]
Balanced accuracy 0.4674133696109414
Compute predictions: 100%|                                                     | 884/884 [02:48<00:00,  5.26it/s]
Balanced accuracy 0.4775601000603386
Compute latents for the evaluation: 100%|                                      | 4252/4252 [06:42<00:00, 10.57it/s]
Compute latents for the evaluation: 100%|                                      | 1056/1056 [01:41<00:00, 10.36it/s]
Compute latents for the evaluation: 100%|                                      | 1770/1770 [02:49<00:00, 10.44it/s]
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$
```

- The results of the analyses are saved in the directory `data/experiments/jump/images/embedding/specificity_target_emb_cv_strat/fold_#` where # is 0,1,2 or 3 respectively
- By default the results are saved in a subdirectory which name is the timestamp it was created
- Rename the directory to `nuclei_regions` as shown in the screenshot below

```
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/image
s/embedding/specificity_target_emb_cv_strat/fold_0$ tree -L 2 .
.
└── nuclei_regions
    ├── best_model_weights.pth
    ├── confusion_matrix_test.png
    ├── confusion_matrix_train.png
    ├── confusion_matrix_val.png
    ├── epoch_0
    ├── fold_0.yml
    ├── logs20231003_112842.log
    ├── plotted_fitting_hist.png
    ├── test
    ├── test_cmatrix.csv
    ├── test_latents.h5
    ├── train_cmatrix.csv
    ├── train_latents.h5
    ├── val_cmatrix.csv
    └── val_latents.h5

3 directories, 13 files
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/image
s/embedding/specificity_target_emb_cv_strat/fold_0$
```

Compute single-cell image embeddings
Time: 9 hours
Size: 10 GB

- Run the script to infer the image embeddings for all 175 train and potential test conditions in the conda environment via

```
conda activate image2reg
python run.py --config
config/image_embedding/specific_targets/extract_latents/extract_latents_
jump_data_resnet_ensemble_specific_targets.yml
```

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ python run.py --config conf
ig/image_embedding/specific_targets/extract_latents/extract_latents_jump_data_re
snet_ensemble_specific_targets.yml
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg$ python run.py --config conf
ig/image_embedding/specific_targets/extract_latents/extract_latents_jump_data_re
snet_ensemble_specific_targets.yml
Compute predictions:  91%|                | 22406/24654 [3:37:09<18:16,  2.05it/s]
```

- The script saves all generated outputs in the directory `data/experiments/jump/images/embedding/extract_latents_from_rohban_trained`
- By default the results are saved in a subdirectory which name is the timestamp it was created
- Copy the content of the timestamp directory into the parent directory `extract_latents_from_rohban_trained` as shown in the screenshot below

```
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/images
/embedding/extract_latents_from_rohban_trained$ ls
confusion_matrix_test.png
confusion_matrix_train.png
confusion_matrix_val.png
extract_latents_jump_data_resnet_ensemble_specific_targets.yml
logs20230223_150829.log
test_cmatrix.csv
test_latents.h5
train_cmatrix.csv
train_latents.h5
val_cmatrix.csv
val_latents.h5
(i2r) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/images
/embedding/extract_latents_from_rohban_trained$
```

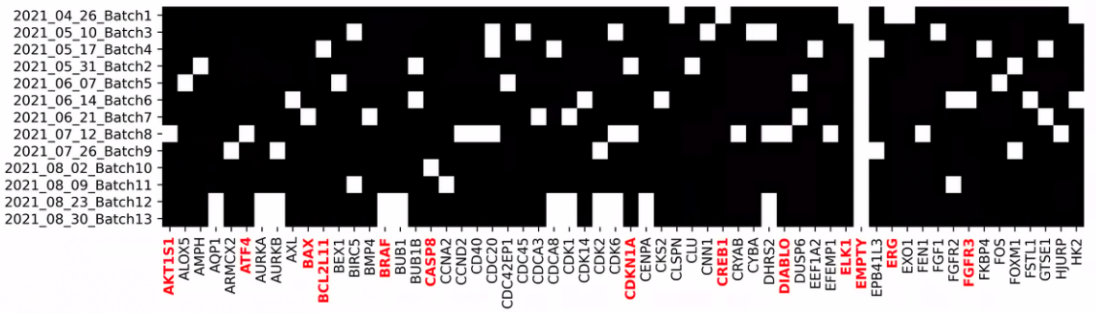
Compute gene perturbation embeddings
Time: 2 hours
Size: 15 GB

- Start the jupyter server in the conda environment via

```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/jump/eda/eda_jump_image_representations.ipynb`
- Run all cells to create e.g the Supplemental Figures S22 and generate the gene perturbation embeddings

```
In [39]: fig, ax = plt.subplots(figsize=[12, 3])
ax = sns.heatmap(
    cond_batch_coocc_matrix.transpose().iloc[:, :60],
    # row_cluster=False,
    # dendrogram_ratio=0.001,
    # cbar_pos=None,
    # figsize=[12, 4],
    ax=ax,
    cbar=False,
    cmap="gray",
)
for tick_label in ax.get_xticklabels():
    tick_text = tick_label.get_text()
    if tick_text in impactful_conditions:
        tick_label.set_color("r")
        tick_label.set_fontweight("bold")
plt.show()
```



- The latter are saved alongside other embeddings in `data/experiments/jump/images/embedding/embeddings` as seen in the screenshot below

```
(base) paysan_d@MPC2838:~$ cd /media/paysan_d/T7/image2reg/data/experiments/jump
/images/embedding/embeddings/
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/image
s/embedding/embeddings$ ls
jump_img_embs_bc.h5  jump_img_embs_insp.h5       jump_morph_profiles.h5
jump_img_embs.h5     jump_morph_profiles_bc.h5   jump_morph_profiles_insp.h5
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/image
s/embedding/embeddings$
```

- The cells also download the morphological profiles for the JUMP-CP data set which will be saved in `data/resources/images/jump/profiles` as seen in the screenshot below

```
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/resources/images/jump/
profiles$ ls
source_4_2021_04_26_Batch1_BR00117035.parquet
source_4_2021_04_26_Batch1_BR00117036.parquet
source_4_2021_04_26_Batch1_BR00117037.parquet
source_4_2021_04_26_Batch1_BR00117038.parquet
source_4_2021_04_26_Batch1_BR00117039.parquet
source_4_2021_04_26_Batch1_BR00117040.parquet
source_4_2021_04_26_Batch1_BR00117041.parquet
source_4_2021_04_26_Batch1_BR00121438.parquet
source_4_2021_04_26_Batch1_BR00121439.parquet
source_4_2021_04_26_Batch1_BR00121537.parquet
source_4_2021_04_26_Batch1_BR00121538.parquet
source_4_2021_04_26_Batch1_BR00121539.parquet
source_4_2021_04_26_Batch1_BR00121540.parquet
source_4_2021_04_26_Batch1_BR00121541.parquet
source_4_2021_04_26_Batch1_BR00121557.parquet
source_4_2021_04_26_Batch1_BR00121558.parquet
source_4_2021_04_26_Batch1_BR00121559.parquet
source_4_2021_04_26_Batch1_BR00121560.parquet
source_4_2021_04_26_Batch1_BR00121561.parquet
source_4_2021_04_26_Batch1_BR00121562.parquet
source_4_2021_04_26_Batch1_BR00121563.parquet
```

- Next, start the jupyter notebook `notebooks/jump/embeddings/analyses_jump_embedding_candidates.ipynb` in the same jupyter session
- Run all cells to generate the input data for the translation analyses
- All generated data will be located in `data/experiments/jump/images/embedding/all_embeddings` as seen in the screenshot below

```
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/image
s/embedding/all_embeddings$ tree
.
├── cmap_signatures.csv
├── mean_batch_embeddings
│   ├── mean_batch_jump_img_embs_bc.csv
│   ├── mean_batch_jump_img_embs.csv
│   ├── mean_batch_jump_img_embs_insp.csv
│   ├── mean_batch_jump_morph_profiles_bc.csv
│   ├── mean_batch_jump_morph_profiles.csv
│   └── mean_batch_jump_morph_profiles_insp.csv
├── mean_embeddings
│   ├── mean_jump_img_embs_bc.csv
│   ├── mean_jump_img_embs.csv
│   ├── mean_jump_img_embs_insp.csv
│   ├── mean_jump_morph_profiles_bc.csv
│   ├── mean_jump_morph_profiles.csv
│   └── mean_jump_morph_profiles_insp.csv
├── mean_rohban_img_embs.csv
└── regulatory_gene_embeddings_using_cmap.csv

2 directories, 15 files
(base) paysan_d@MPC2838:/media/paysan_d/T7/image2reg/data/experiments/jump/image
s/embedding/all_embeddings$
```

Performance evaluation

Run gridsearch and analyze the results

Time: 3 hours

Size: 3 GB

- Start the jupyter server in the conda environment via

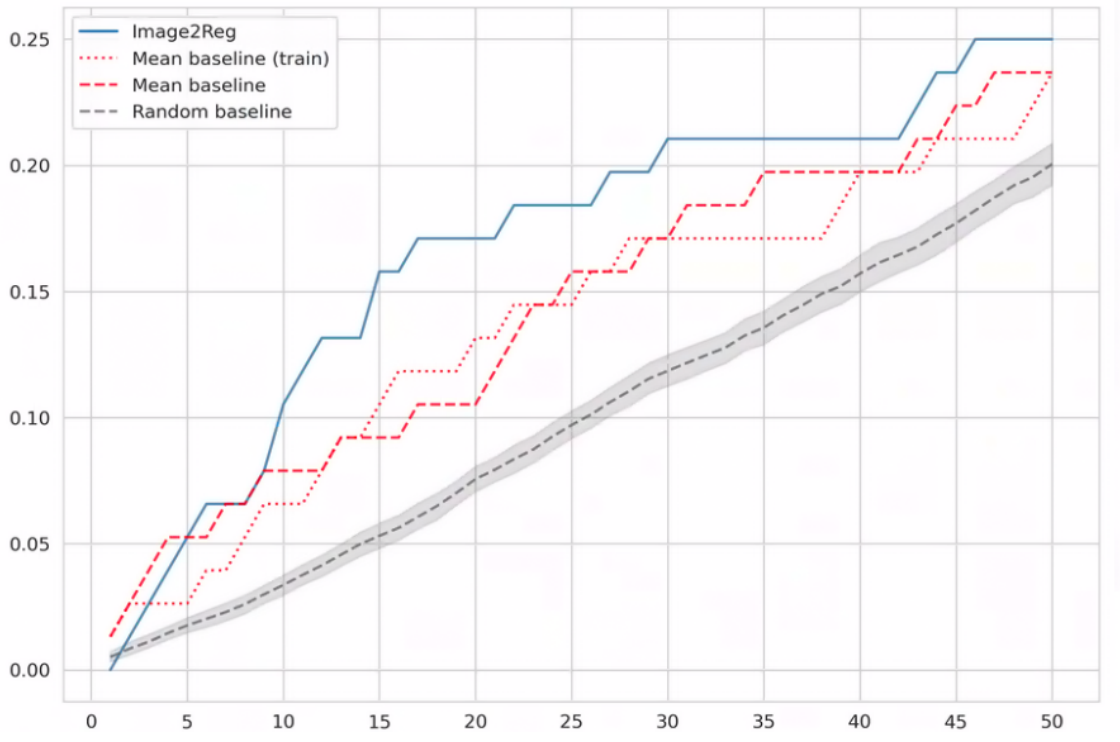
```
conda activate image2reg
jupyter notebook
```

- Start the notebook `notebooks/jump/translation/jump_translation_prediction_final.ipynb`
- Run all cells to perform the complete translation analysis and e.g.generate Fig. 4C as seen in the screenshot below

In [75]:

```
fig, ax = plot_translation_performance(
    all_knn_results.loc[all_knn_results.loc[:, "Model"] == "Image2Reg"],
    hue="Model",
    ymax=0.3,
    title="",
    random_baseline=cc_random_baseline_knn_accs,
    mean_baseline=cc_mean_baseline_knn_accs,
    mean_train_baseline=cc_train_mean_baseline_knn_accs,
    figsize=[10, 7],
    cmap=None,
    style=None,
    style_order=None,
    param_title=None,
    legend_title="",
    alpha=0.8,
)
# ax.set_title("Performance for mean embeddings/profiles")
# ax.set_ylabel("kNN accuracy")
# ax.set_xlabel("k nearest neighbors")
ax.set_xlabel("")
ax.set_ylabel("")
```

Out[75]: `Text(0, 0.5, '')`



This concludes the reproduction of all results presented in our study from scratch.