HYDRA: Heterogeneity through discriminative analysis

Erdem Varol
erdem.varol@uphs.upenn.edu
software@cbica.upenn.edu

April 3, 2018

1 Introduction

This software performs clustering of heterogenous disease patterns within patient group. The clustering is based on seperating the patient imaging features from the control imaging features using a convex polytope classifier. Covariate correction can be performed optionally.

2 Method

There is ample evidence for the heterogeneous nature of diseases. For example, Alzheimer's Disease, Schizophrenia and Autism Spectrum Disorder are typical disease examples that are characterized by high clinical heterogeneity, and likely by heterogeneity in the underlying brain phenotypes. Parsing this heterogeneity as captured by neuroimaging studies is important both for better understanding of disease mechanisms, and for building subtype-specific classifiers. However, few existing methodologies tackle this problem in a principled machine learning framework.

In this work, we developed a novel non-linear learning algorithm for integrated binary classification and subpopulation clustering. Non-linearity is introduced through the use of multiple linear hyperplanes that form a convex polytope that separates healthy controls from pathologic samples. Disease heterogeneity is disentangled by implicitly clustering pathologic samples through their association to single linear sub-classifiers. (Figure 1)

For more information, please refer to:

```
Varol, Erdem, Aristeidis Sotiras, Christos Davatzikos.
"HYDRA: Revealing heterogeneity of imaging and genetic patterns
through a multiple max-margin discriminative analysis framework."
 NeuroImage 145 (2017): 346-364.
```

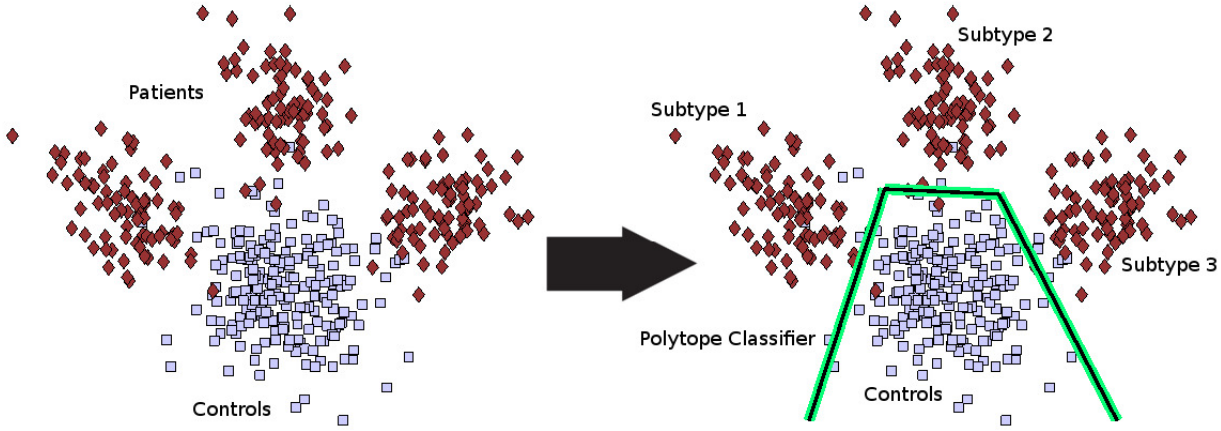

Figure 1: Schematic illustration of the HYDRA method. Controls are separated from the patients using a convex polytope decision boundary. The association of the patients with the sides of the convex polytope induces a clustering assignment.

3 Testing & Installation

This software has been primarily implemented in MATLAB for Linux operating systems.

3.1 Requirements

1. Matlab optimization toolbox
2. Matlab version >2014

3.2 Installation

Hydra can be run directly in a matlab environment without compilation.

OPTIONAL

If the user wants to run hydra as a standalone executable, then it must be compiled as following (using the additionally obtained matlab compiler "mcc"):

Run the following command in a MATLAB environment:

```
mcc -m hydra.m
```

3.3 Testing

We provided a test sample in the test folder.

To test in matlab enviroment, use the command:

```
hydra('-i','test.csv','-o','.','-z','test_covar.csv','-k',3,'-f',3)
```

To test in command line using the compiled executable, use the command:

```
hydra -i test.csv -o . -z test_covar.csv -k 3 -f 3
```

This runs a HYDRA experiment which may take a few minutes. The test case contains a subset of a functional MRI study dataset by T. Satterwaithe comprising 100 subjects and their functional ROI's. The output is the clustering labels of the input subjects (only patients are clustered) at varying clustering levels. Also, the clustering stability at varying levels is output to show the rationale for choosing the clustering level.

3.4 Test verification

Pre-computed HYDRA results have been included in directory "`Pre_computed_test_results`". The user may verify that their test results match the pre-computed results to confirm proper set-up. If the clustering occurred properly, ARI for clustering level `k=3` should be greater than that of clustering level `k=2`.

4 Usage

```
function returns estimated subgroups by hydra for clustering
configurations ranging from K=1 to K=10, or another specified range of
values. The function returns also the Adjusted Rand Index that was
calculated across the cross-validation experiments and comparing
respective clustering solutions.

INPUT

REQUIRED
[--input, -i] : .csv file containing the input features. (REQUIRED)
             every column of the file contains values for a feature, with
             the exception of the first and last columns. We assume that
             the first column contains subject identifying information
             while the last column contains label information. First line
             of the file should contain header information. Label
             convention: -1 -> control group - 1 -> pathological group
             that will be partioned to subgroups
[--outputDir, -o] : directory where the output from all folds will be saved (REQUIRED)

OPTIONAL

[--covCSV, -z] : .csv file containing values for different covariates, which
          will be used to correct the data accordingly (OPTIONAL). Every
          column of the file contains values for a covariate, with the
          exception of the first column, which contains subject
          identifying information. Correction is performed by solving a
          solving a least square problem to estimate the respective
          coefficients and then removing their effect from the data. The
          effect of ALL provided covariates is removed. If no file is
          specified, no correction is performed.

NOTE: featureCSV and covCSV files are assumed to have the subjects given
      in the same order in their rows

[--c, -c] : regularization parameter (positive scalar). smaller values produce
    sparser models (OPTIONAL - Default 0.25)
[--reg_type, -r] : determines regularization type. 1 -> promotes sparsity in the
           estimated hyperplanes - 2 -> L2 norm (OPTIONAL - Default 1)
[--balance, -b] : takes into account differences in the number between the two
          classes. 1-> in case there is mismatch between the number of
          controls and patient - 0-> otherwise (OPTIONAL - Default 1)
[--init, -g] : initialization strategy. 0 : assignment by random hyperplanes
       (not supported for regression), 1 : pure random assignment, 2:
       k-means assignment, 3: assignment by DPP random
       hyperplanes (default)
```

```
[--iter, -t] : number of iterations between estimating hyperplanes, and cluster
       estimation. Default is 50. Increase if algorithms fails to
       converge
[--numconsensus, -n] : number of clustering consensus steps. Default is 20.
               Increase if algorithm gives unstable clustering results.
[--kmin, -m] : determines the range of clustering solutions to evaluate
            (i.e., kmin to kmax). Default  value is 1.
[--kmax, -k] : determines the range of clustering solutions to evaluate
            (i.e., kmin to kmax). Default  value is 10.
[--kstep, -s] : determines the range of clustering solutions to evaluate
            (i.e., kmin to kmax, with step kstep). Default  value is 1.
[--cvfold, -f]: number of folds for cross validation. Default value is 10.
[--vo, -j] : verbose output (i.e., also saves input data to verify that all were
     read correctly. Default value is 0
[--usage, -u]  Prints basic usage message.
[--help, -h]  Prints help information.
[--version, -v]  Prints information about software version.

OUTPUT:
CIDX: sub-clustering assignments of the disease population (positive
      class).
ARI: adjusted rand index measuring the overlap/reproducibility of
     clustering solutions across folds

NOTE: to compile this function do
mcc -m  hydra.m


EXAMPLE USE (in matlab)
hydra('-i','test.csv','-o','.','-k',3,'-f',3);
EXAMPLE USE (in command line)
hydra -i test.csv -o . -k 3 -f 3
```

4.1 Running "HYDRA"

Here is a brief introduction to running HYDRA. For a complete list of parameters, see `--help` option.

To run this software, you will need an input csv file, with the following mandatory fields in the following column order: (Column 1) `ID`: ID for subject (Column 2—(last minus 1)) `features`: features to be used for clustering (Column (last)) `groups`: label whether the subject is control (-1) or patient (1)

NOTE: Controls must be strictly `-1` and patients must be `1` label. NOTE: Label headers names are not strict.

An example input csv file looks as following:

```
ID,        feature_1,    feauture_2,    feature_3,    group
subject_1,         5,            1,          79.3,         -1
subject_2,        10,            1,          71.4,         1
subject_3,         3,            1,          82.7,         -1
```

Optionally, you can provide a covariate file that will be used to remove covariate effects from imaging features before HYDRA analysis. The covariate file has the following format: (Column 1) `ID`: ID for subject (Column 2—(last)) `covariates`: covariates of subjects

An example covariate csv file looks as following:

```
ID,        age,        sex
subject_1,    29,        1
subject_2,    35,        1
subject_3,    51,        0
```

If you install the package successfully, there will be two ways of running HYDRA:

1. Running HYDRA in a matlab environment, a simple example:

```
hydra('-i','test.csv','-o','.','-z','test_covar.csv','-k',3,'-f',3)
```

2. Running matlab compiled HYDRA executables in the command line, a simple example:

```
hydra -i test.csv -o . -z test_covar.csv -k 3 -f 3
```

4.2 Output

The software returns, `HYDRA_results.mat` in the specified output directory. This `mat` file stores the following variables:

- `CIDX` - clustering indices for subjects (rows) at varying levels (columns)
- `ARI` - adjusted rand index of clustering at varying levels, clustering level at the highest ARI should be selected
- `ID` - subject ID of rows

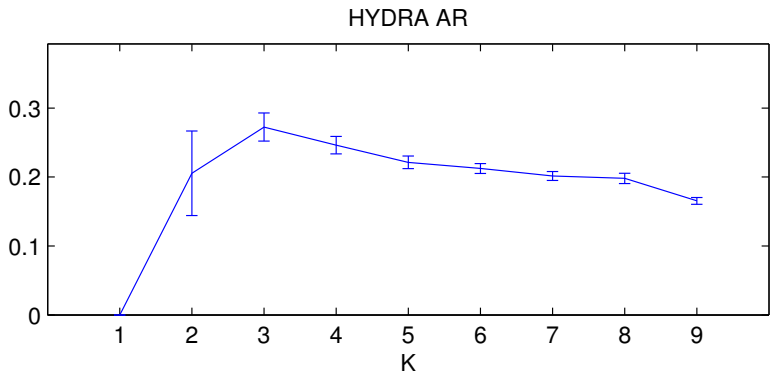

Figure 2: The adjusted rand index for the Alzheimer's Disease dataset at varying clustering levels k. The level of clustering that produces the most stable results is $k = 3$.

4.3 Determining the clustering level: k

The user can set the range and step size of clustering levels to be attempted using the `kmin`, `kmax` and `kstep` parameters. The output file `HYDRA_results.mat` stores the variable `ARI` that stores the clustering stability for the range of values of k. The k that achieved the highest ARI (adjusted rand index) should be used (Figure 2).

5 Citation

If you find this software useful, please cite:

```
Varol, Erdem, Aristeidis Sotiras, Christos Davatzikos.
"HYDRA: Revealing heterogeneity of imaging and genetic patterns
through a multiple max-margin discriminative analysis framework."
 NeuroImage 145 (2017): 346-364.
```

6 Licensing

See `https://www.med.upenn.edu/sbia/software-agreement.html` or `COPYING.txt` file.