**Running CRF model on Yeti**

**CRF codebase**

The original version from Kui Tang in Jebara’s group is at <https://github.com/kuitang/fwmatch-public>. This particular repo might not be up-to-date.

My modified version is at <https://github.com/hanshuting/graph_ensemble>. This repo is currently private, but I might be able to give you access upon request. After downloading the codebase to your yeti directory, compile the dependencies as described in the readme file.

The codebase is organized as following: all experiments should be under expt/ directory; mvc/ has my matlab code to process the results; src/ has all source code needed to run the model; thirdparty/ has some third-party packages needed for the model.

**Working with Yeti**

See documentation at <https://wikis.cuit.columbia.edu/confluence/display/rcs/Yeti+HPC+Cluster+User+Documentation>.

**Data format**

A binary spike matrix should be stored in a .mat file, under the variable name “data”. “data” needs to be number of frame by number of neuron. The .mat file should be named as experiment\_condition.mat, for example, given the m21\_d2\_vis dataset, the “experiment” is “m21\_d2\_vis”, the condition could be something like “high\_add\_neuron”, therefore the .mat file should be named as m21\_d2\_vis\_high\_add\_neuron.mat. This allows you to run multiple files that are originated from the same dataset but processed differently (high activity frame vs all frames, visual stimulations only vs all frames, no add neuron vs add neuron model, etc.) at the same time. In my case, the data file is saved in ~/data/luis/[filename].

**Running CRF model - An example**

1. Upload a data file $DATAPATH/test\_1.mat (I will prepare a test dataset so that people can try with the same dataset).

2. Go to expt directory: cd ~/src/fwMatch-darpa/expt

3. Make a template dir: cp -r m21\_d2\_vis\_template/ test\_template/

Most files in this template directory don’t matter; they’re here only for historical reasons. The only files that matter are create\_config\_files.m and merge\_all\_models.m. I will clean up the codebase in the future.

4. Modify create\_script.pl accordingly:

$EXPT\_NAME = "test";

@EE = ("1"); # note you can put multiple experiments here as (“1”,”2”,”3”);

$MODEL\_TYPE = "loopy";

$DATA\_DIR = "/vega/brain/users/sh3276/data/luis";

Between line 42 and 59, modify the email address to yours, and yeti\_user to your UNI. You can also adjust the model parameter test ranges here.

5. Start an interactive job:

qsub -I -q interactive -W group\_list=yetibrain -l walltime=00:30:00,mem=2000mb

6. Run create\_script by: ./create\_script.pl

This script will create a working directory based on the template directory; the working directory will be named as experiment\_condition\_loopy/ (in this case, test\_1\_loopy/). Then, it will write the following files to your experiment directory: get\_real\_data.m (for loading data), write\_config\_for\_loopy.m (configuration file template for loopy models), write\_config\_for\_tree.m (configuration file template for tree models). The next thing it does is to start matlab, and write configuration files for each parameter combination by executing the function create\_config\_files.m. This will generate 30 files like config1.m under the directory. Finally, it brings you back to the expt/ dir.

If you want to modify your YETI submission script settings (for example, letting YETI sending you an email when the job is done), modify create\_config\_files.m in the template directory before running create\_script.pl. Between line 115 and 143 is where you look for YETI-related information. For Luis’s datasets, 12 hours walltime and 8G memory is enough.

7. Go to working dir and start job:

cd test\_1\_loopy/

./start\_jobs.sh

8. Check your job status by: qstat -t -u <UNI>

9. Once the job is done running, start an interactive job, and do the following:

matlab -nodesktop -nosplash -nodisplay

addpath(genpath(‘~/src/fwMatch-darpa’));

cd ~/expt/test\_1\_loopy/

merge\_all\_models;

save\_best\_model;

Typing best\_model.theta should display the model parameters. Take a note of s\_lambda, p\_lambda and density if you want to run shuffled controls of this dataset.

A model\_collection.mat file should be saved under results/.

Finally, exit matlab and finish the interactive job:

exit;

logout

Before running save\_best\_model for the first time, please change the save path to your desired path. This file is located at fwMatch-darpa/mvc/scripts/save\_best\_model.m. Then a file named test\_1\_loopy\_best\_model\_full.mat should be saved this path.

**Running shuffled controls - An example**

1. Go to expt directory: cd ~/src/fwMatch-darpa/expt

2. Make a template dir: cp -r shuffled\_m21\_d2\_vis\_template/ shuffled\_test\_template/

3. Modify create\_shuffle\_script.pl accordingly:

$EXPT\_NAME = "test";

@EE = ("1");

$MODEL\_TYPE = "loopy";

$DATA\_DIR = "/vega/brain/users/sh3276/data/luis";

@DENSITY = (0.29); # put your best density value here

@S\_LAMBDA = (1.8206e-04); # put your best s\_lambda here

@P\_LAMBDA = (56.2341); # put your best p\_lambda here

$NSHUFFLE = 100;

4. Start an interactive job, and run: ./create\_shuffle\_script.pl

After this step, you can finish the interactive job by typing logout.

5. Go to your working directory, and run the job that generates shuffled dataset first:

cd shuffled\_test\_1\_loopy/

./shuffle\_start\_job.sh

6. Monitor this job (usually it’s done within an hour), once it’s finished, go to working directory, and start running CRF models on shuffled data:

./start\_job.sh

7. When all jobs are done, start an interactive job, and start matlab:

matlab -nodesktop -nosplash -nodisplay

addpath(genpath(‘~/src/fwMatch-darpa’));

cd ~/expt/shuffled\_test\_1\_loopy/

merge\_all\_models;

save\_shuffled\_model;

exit;

Again, please change the save path in save\_shuffled\_model.m to your desired path before running it for the first time.