EXAMPLE

REPTILE

- Regulatory Element Prediction based on TIssue-specific Local Epigenetic marks

https://github.com/yupenghe/REPTILE

REPTILE is a tool to identify the precise location of enhancers by integrating histone modification data and base-resolution DNA methylation profiles.

Please contact Yupeng He for for feedbacks, questions or bugs.

OVERVIEW

The document includes an example of using REPTILE to predict enhancers in mouse tissues. This example is based on the exact dataset used in the REPTILE manuscript. First, REPTILE will be run to learn an enhancer model based on the data in mouse embryonic stem cells (mESCs). In the training dataset, EP300 binding sites are used as representative active enhancers and the promoters regions along with random genomic intervals are used as inative instances. In the next step, the enhancer model learned in mESCs will be applied to predict the enhancer activity of 545 genomic elements in the heart tissue from mouse embryo at E11.5 developmental stages (E11_5_HT). These elements are from VISTA enhancer browser and were experimentally validated using transgenic reporter assay. We will compare the predictions with experimental results to evaluate the accuracy of REPTILE. Lastly, we will use REPTILE to generate enhancer predictions across the genome.

The dataset used in the example includes DNA methylation (Meth) and 6 histone modifications (H3K4me1, H3K4me2, H3K4me3, H3K27ac, H3K27me3 and H3K36me3) of in total nine samples. These samples are mESCs and 8 mouse tissues from embryo at E11.5 developmental stage:

- E11_5_HT: E11.5 heart
- E11_5_LM: E11.5 limb
- E11_5_FB: E11.5 forebrain
- E11_5_MB: E11.5 midbrain
- E11_5_HB: E11.5 hindbrain
- E11_5_NT: E11.5 neural tube
- E11_5_CF: E11.5 embryonic facial prominence
- E11_5_LV: E11.5 liver

In addition, the dataset contains 40,000 genomic regions for training REPTILE and 545 genmoic regions to evaluation its prediction accurary. Differetially methylated regions (DMRs), which are needed to improve the resolution of REPTILE predictions, were called by comparing the DNA methylation profile of all 9 samples.

REQUIREMENTS

It is recommended to run this example in a computer cluster (server) because of the runtime and the requirement for memory. Finishing the entire example requires minimum 8 Gb memory (single CPU) and 60 Gb space on the hard drive. Although 8 Gb memory is enough, to achieve the memory requirement requires only one CPU to use and the runtime will be >20 hours. Roughly, the peak of memory usage is the product of number of processors (`8` in the code below) and 8 Gb. Each processor will take around 8 Gb memory. If the total memory requirement cannot be met, it is better to reduce the number of processors to use by change the value passed to the `-p` option when running `REPTILE_compute_score.R` and `REPTILE_preprocess.py` .

If the server meets the requirement of 64 Gb memory (8 Gb x 8 CPU to use) and 60 Gb space, you can run the example (which will take ~3h) by simply doing:

```
cd example/
sh run_example.sh > log 2> err
```

Otherwise, you will need to change the value assigned to `num_procs` in `run_example.sh` to reduce the need of memory before running the script.

A SIMPLE EXAMPLE

A simple example is also provided. 1 Gb hard drive space and 1 Gb memory are needed to run it.

```
cd simple_example/
sh run_simple_example.sh > log 2> err
```

It takes around 20min using single CPU. You may change the `num_procs` value in `run_simple_example.sh` script to run it with multiple CPUs. Compared with the complete example, the simple example includes only the data on chromosome 19. However, the output is less meaningful since only a subset of data is used.

STEP BY STEP

Download example data

Create a folder to run the example.

```
mkdir REPTILE_example/
cd REPTILE_example/
mkdir -p tmp/ results/
```

Download and unzip the data needed for running the complete example.

```
wget neomorph.salk.edu/yupeng/share/REPTILE_example_data.tar
tar xf REPTILE_example_data.tar
rm REPTILE_example_data.tar
```

For the simple example, the code changes to:

```
wget neomorph.salk.edu/yupeng/share/REPTILE_simple_example_data.tar
tar xf REPTILE_simple_example_data.tar
rm REPTILE_simple_example_data.tar
```

Other steps are the same.

Description of example data

In the newly created `REPTILE_example/` folder, you should see the below files/folders.

- `data/bw/` is folder contains the bigWig files of all epigenetic marks of all nine samples
- `data/data_info_mESC_E11_5.tsv` is the data info file indicating the samples and marks involved as well as the corresponding bigWig files.
- `data/DMR_CG_mESC_E11_5_ext150.bed` contains genomic coordinates of all DMRs.
- `data/mm10_chrLen.tsv` has the length of each chromosome in mouse mm10 reference.
- `data/training_data/` and `data/test_data/` contain the data (files) needed for training and accuracy evaluation respectively.
- `data/training_data/mESC_region_for_train.bed` and `data/test_data/vista_enhancer_mm10_mm.bed` contain the genomic coordinates of regions for training and regions for evaluating REPTILE accuracy.
- `data/training_data/mESC_region_for_train_label.tsv` and `data/test_data/vista_enhancer_state.tsv` are the label files for regions contained in the two files above.

Training REPTILE

In this step, we train REPTILE on data of mESCs. First, we preprocess the training data using 8 CPUs.

```
REPTILE_preprocess.py \
    data/data_info_mESC_E11_5.tsv \
    data/training_data/mESC_region_for_train.bed \
    tmp/training_region \
    -d data/DMR_CG_mESC_E11_5_ext150.bed \
    -p 8
```

The parameter, `tmp/training_region`, specifies the prefix of output files. Two output files will be generated, `tmp/training_region.region_with_epimark.tsv` and `tmp/training_region.DMR_with_epimark.tsv`, containing the epigenomic signatures of query regions and DMRs, respectively.

Then, `REPTILE_train.R` is run to learn an enhancer model from the training dataset.

```
REPTILE_train.R \
    -i data/data_info_mESC_E11_5.tsv \
    -a tmp/training_region.region_with_epimark.tsv \
    -d tmp/training_region.DMR_with_epimark.tsv \
    -l data/training_data/mESC_region_for_train_label.tsv \
    -s mESC \
    -o tmp/REPTILE_model
```

The parameters of the enhancer model are stored in the output file `tmp/REPTILE_model.reptile`.

Predict the enhancer activity of VISTA elements

With the trained enhancer model, we then apply REPTILE to predict the enhancer activity in test dataset, which contain 545 elements from VISTA enhancer browser. Simiarly, we first preprocess the test data.

```
REPTILE_preprocess.py \
    data/data_info_mESC_E11_5.tsv \
    data/test_data/vista_enhancer_mm10_mm.bed \
    tmp/test_region \
    -d data/DMR_CG_mESC_E11_5_ext150.bed
```

Next, REPTILE calculates enhancer confidence score for each query region and DMR based on the enhancer model. Note that `-w` option is used in order to get the combined score of each query region (see below).

```
## Generating enhancer scores
REPTILE_compute_score.R \
    -i data/data_info_mESC_E11_5.tsv \
    -m tmp/REPTILE_model.reptile \
    -d tmp/test_region.DMR_with_epimark.tsv \
    -a tmp/test_region.region_with_epimark.tsv \
    -s E11_5_HT \
    -o results/E11_5_HT_pred \
    -w
```

Three output files will be in `results/`.

- `E11_5_HT_pred.D.bed`: combined score of query reigons. For each query region, combined score is defined as the maximum of score of the query region and the DMRs overlapping with it.
- `E11_5_HT_pred.R.bed`: score of query regions
- `E11_5_HT_pred.DMR.bed`: score of DMRs

Last, we evaluate the prediction result by comparing it with experimental data (`data/test_data/vista_enhancer_state.tsv`).

```
## Evaluate the prediction results
echo -n "E11_5_HT "
echo \
`REPTILE_evaluate_prediction.R \
    -p results/E11_5_HT_pred.D.bed \
    -s E11_5_HT \
    -l data/test_data/vista_enhancer_state.tsv`
done
```

If the code executes correctly, the result should be like:

```
Sample AUROC AUPR top5 top10 top20
E11_5_HT 0.8351515 0.5800887 4 8 17
```

AUROC is short for The area under the receiver operating characteristic curve, which AUPR is short for area under the precision-recall curve. They are two metrics of prediction accuracy. "top5", "top10" and "top20" are the percentage of true positives in the top 5, 10 and 20 predictions respectively.

Generate putative enhancers across genome

In the last part of the example, we use REPTILE to generate genome-wide enhancer predictions (calls). We first geenrate 2kb sliding windows across the entire mouse genome using bedtools. Then, preprocessing is conducted to get the epigenomic signatures of the sliding windows and DMRs. Note that `-g` option is used.

```
## Generate sliding windows across mouse genome
bedtools makewindows -w 2000 -s 100 -g data/mm10_chrLen.tsv |awk '{print $_"\tbin_"i++}' > tmp/mm10_w2kb_s100bp.bed

## Preprocessing
REPTILE_preprocess.py \
    data/data_info_mESC_E11_5.tsv \
    tmp/mm10_w2kb_s100bp.bed \
    tmp/mm10_w2kb_s100bp \
    -d data/DMR_CG_mESC_E11_5_ext150.bed \
    -n 12 \
    -p 8 \
    -g
```

Next, enhancer score is calculated for each query region and each DMR.

```
## Generating enhancer scores
REPTILE_compute_score.R \
    -i data/data_info_mESC_E11_5.tsv \
    -m tmp/REPTILE_model.reptile \
    -d tmp/mm10_w2kb_s100bp.DMR_with_epimark \
    -a tmp/mm10_w2kb_s100bp.region_with_epimark \
    -s E11_5_HT \
    -o tmp/E11_5_HT_pred \
    -p 8 \
    -n 12
```

Based on the score of query regions and DMRs, REPTILE identifies putative elements in the genome given score cutoff, 0.5.

```
## Call putative enhancers based on the scores
REPTILE_call_enhancer.py \
    tmp/E11_5_HT_pred.R.bed \
    -d tmp/E11_5_HT_pred.DMR.bed \
    -p 0.5 \
    -o results/enhancer_E11_5_HT.bed
done
```

The output file `results/enhancer_E11_5_HT.bed` stores the enhancer predictions.