Basenji deeplearning model

---

1. All data and relevant code

As they show (https://github.com/calico/basenji/tree/master/manuscripts/saluki),

- the manuscript can be found here:

  https://genomebiology.biomedcentral.com/articles/10.1186/s13059-022-02811-x
- all the models (train, validate, test files) can be found here (in datasets/deeplearning/train_gru):

  https://doi.org/10.5281/zenodo.6326409
- Manual:

  https://github.com/calico/basenji/tree/master/tutorials

---

2. Preprocess

Download the Model

```
!git clone https://github.com/calico/basenji.git
```

Make sure the tensorflow version == 2.8.0:

(tf version 2.1.7 doesn't work)

```
import tensorflow as tf
print(tf.__version__)
```

Install required packages:

```
!pip install intervaltree
!pip install pysam
!pip install pyBigWig
```

Download test data:

```
! wget
https://storage.googleapis.com/basenji_tutorial_data/heart_l131k.tgz
! tar -xzvf heart_l131k.tgz
```

If contains:

contigs.bed statistics.json tfrecords
sequences.bed targets.txt

Or users can generate as follows:

```
./basenji_data.py -s .1 -g data/unmap_macro.bed -l 131072 --local -o
data/heart_l131k_generate -p 8 -t .1 -v .1 -w 128 data/hg19.ml.fa
data/heart_wigs.txt
```

Input should be:

- BigWig coverage tracks
- Genome FASTA file

`-g` : Dodge large-scale unmappable regions like assembly gaps

- If can be downloaded from https://github.com/calico/basenji/raw/master/tutorials/data/unmap_macro.bed

Please note that output_dir should be empty.

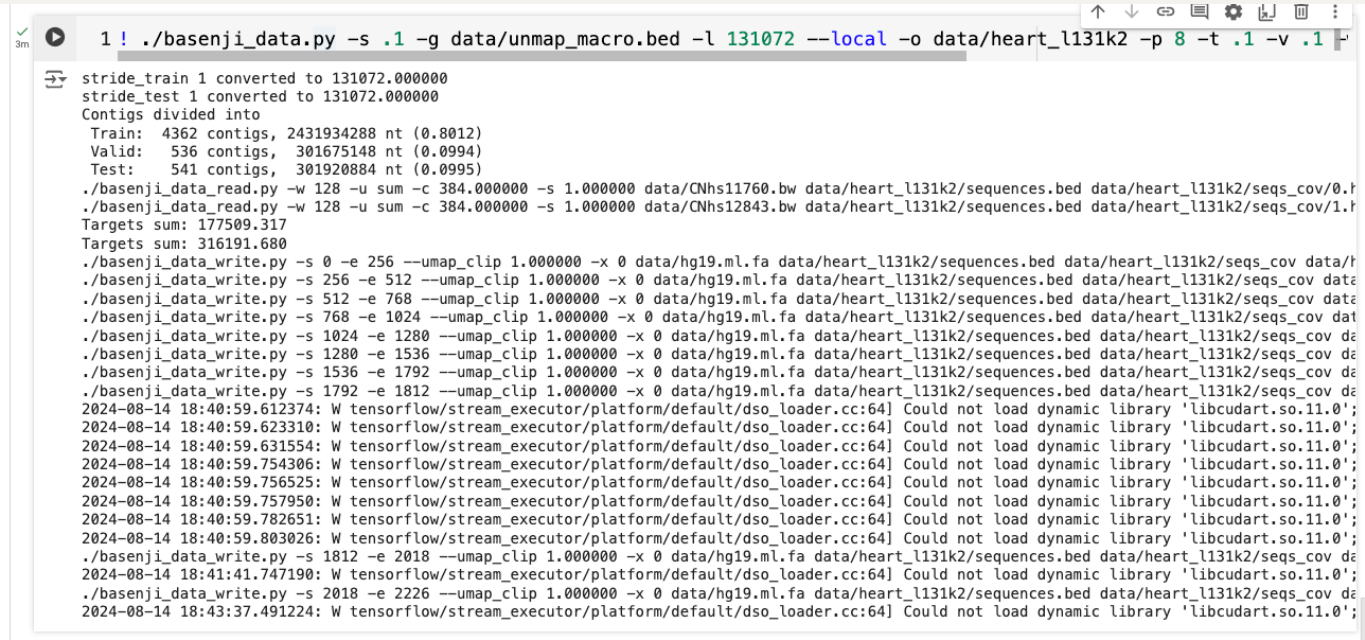
The `heart_wigs.txt` can be generated as follows (containing bigwig files' information):

```
lines =
[['index','identifier','file','clip','sum_stat','description']]
lines.append(['0', 'CNhs11760', 'data/CNhs11760.bw', '384', 'sum',
'aorta'])
lines.append(['1', 'CNhs12843', 'data/CNhs12843.bw', '384', 'sum',
'artery'])
lines.append(['2', 'CNhs12856', 'data/CNhs12856.bw', '384', 'sum',
'pulmonic_valve'])

samples_out = open('heart_wigs.txt', 'w')
for line in lines:
    print('\t'.join(line), file=samples_out)
samples_out.close()
```

Please make sure `basenji_data_write.py` and `basenji_data_read.py` is in the same folder as `basenji_data.py`.

Generation process:



---

3. Train

Input:

- model parameters
- data folder (generated from step 2)
- output dir

Output:

- Model_check.h5
- Model_best.h5

Code:

```
! basenji/basenji_train.py \
    -o test_models/ basenji/tutorials/models/params_small.json
basenji/data/heart_l131k
```

It takes several hours to train.

Framework (Part) :

```
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
sequence (InputLayer)           [(None, 131072, 4)]  0           []

stochastic_reverse_complement   ((None, 131072, 4),  0          ['sequence[0][0]']
(StochasticReverseComplement)   ())

stochastic_shift (StochasticSh  (None, 131072, 4)    0          ['stochastic_reverse_complement[0
ift)                                                            ][0]']

tf.nn.gelu (TFOpLambda)         (None, 131072, 4)    0          ['stochastic_shift[0][0]']

conv1d (Conv1D)                 (None, 131072, 64)   3840       ['tf.nn.gelu[0][0]']

batch_normalization (BatchNorm  (None, 131072, 64)  256         ['conv1d[0][0]']
alization)

max_pooling1d (MaxPooling1D)    (None, 16384, 64)    0          ['batch_normalization[0][0]']

tf.nn.gelu_1 (TFOpLambda)       (None, 16384, 64)    0          ['max_pooling1d[0][0]']

conv1d_1 (Conv1D)               (None, 16384, 64)    20480      ['tf.nn.gelu_1[0][0]']

batch_normalization_1 (BatchNo  (None, 16384, 64)   256         ['conv1d_1[0][0]']
rmalization)

max_pooling1d_1 (MaxPooling1D)  (None, 4096, 64)     0          ['batch_normalization_1[0][0]']

tf.nn.gelu_2 (TFOpLambda)       (None, 4096, 64)     0          ['max_pooling1d_1[0][0]']
```

Training process:

```
==================================================================================================
Total params: 111,011
Trainable params: 109,235
Non-trainable params: 1,776
__________________________________________________________________________________________________
None
model_strides [128]
target_lengths [1024]
target_crops [0]
No checkpoints found.
Successful first step!
Epoch 0 - 705s - train_loss: 0.4166 - train_r: 0.1858 - train_r2: 0.0268 - valid_loss: 0.3851 - valid_r: 0.2541 - valid_r2: 0.0612 - best!
Epoch 1 - 736s - train_loss: 0.3631 - train_r: 0.2620 - train_r2: 0.0671 - valid_loss: 0.3713 - valid_r: 0.2986 - valid_r2: 0.0603 - best!
Epoch 2 - 749s - train_loss: 0.3487 - train_r: 0.2995 - train_r2: 0.0896 - valid_loss: 0.3975 - valid_r: 0.2967 - valid_r2: -0.0263
Epoch 3 - 746s - train_loss: 0.3502 - train_r: 0.3060 - train_r2: 0.0935 - valid_loss: 0.3592 - valid_r: 0.3108 - valid_r2: 0.0693 - best!
Epoch 4 - 714s - train_loss: 0.3405 - train_r: 0.3404 - train_r2: 0.1152 - valid_loss: 0.3459 - valid_r: 0.3631 - valid_r2: 0.1292 - best!
Epoch 5 - 741s - train_loss: 0.3441 - train_r: 0.3372 - train_r2: 0.1137 - valid_loss: 0.3726 - valid_r: 0.3235 - valid_r2: 0.0648
Epoch 6 - 706s - train_loss: 0.3371 - train_r: 0.3655 - train_r2: 0.1327 - valid_loss: 0.3414 - valid_r: 0.3760 - valid_r2: 0.1310 - best!
Epoch 7 - 729s - train_loss: 0.3288 - train_r: 0.4034 - train_r2: 0.1604 - valid_loss: 0.3426 - valid_r: 0.4272 - valid_r2: 0.1553 - best!
Epoch 8 - 733s - train_loss: 0.3291 - train_r: 0.4072 - train_r2: 0.1632 - valid_loss: 0.3350 - valid_r: 0.3990 - valid_r2: 0.1578
Epoch 9 - 746s - train_loss: 0.3228 - train_r: 0.4522 - train_r2: 0.1987 - valid_loss: 0.3497 - valid_r: 0.4133 - valid_r2: 0.1283
Epoch 10 - 737s - train_loss: 0.3204 - train_r: 0.4724 - train_r2: 0.2175 - valid_loss: 0.3368 - valid_r: 0.4205 - valid_r2: 0.1748
Epoch 11 - 742s - train_loss: 0.3191 - train_r: 0.4976 - train_r2: 0.2380 - valid_loss: 0.3327 - valid_r: 0.4675 - valid_r2: 0.2163 - best!
Epoch 12 - 705s - train_loss: 0.3117 - train_r: 0.5268 - train_r2: 0.2649 - valid_loss: 0.3399 - valid_r: 0.4624 - valid_r2: 0.1813
Epoch 13 - 696s - train_loss: 0.3135 - train_r: 0.5274 - train_r2: 0.2686 - valid_loss: 0.3349 - valid_r: 0.4870 - valid_r2: 0.2336 - best!
Epoch 14 - 688s - train_loss: 0.3127 - train_r: 0.5712 - train_r2: 0.3112 - valid_loss: 0.3440 - valid_r: 0.4824 - valid_r2: 0.1973
Epoch 15 - 700s - train_loss: 0.3110 - train_r: 0.6091 - train_r2: 0.3520 - valid_loss: 0.3327 - valid_r: 0.5102 - valid_r2: 0.2473 - best!
```

4. Test

```
! basenji/basenji_test.py --ai 0,1,2 -o test_output/heart_test --rc --shifts "1,0,-1" \
 basenji/tutorials/models/params_small.json test_models/model_best.h5 basenji/data/heart_l131k
```

```
=================================================================================================
Total params: 111,011
Trainable params: 109,235
Non-trainable params: 1,776
_________________________________________________________________________________________________
None
model_strides [128]
target_lengths [1024]
target_crops [0]
45/45 [==============================] - 104s 2s/step - loss: 0.3224 - pearsonr: 0.5631 - r2: 0.3040

Test Loss:         0.32242
Test PearsonR:     0.56310
Test R2:           0.30398
```

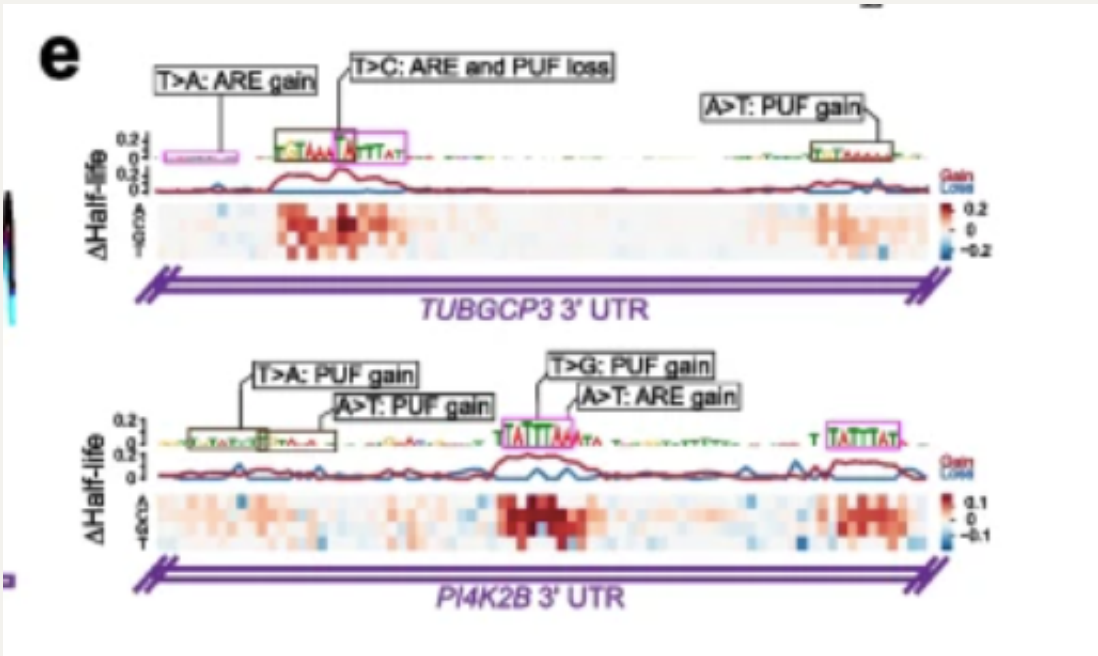
acc.txt:

index	pearsonr	r2	identifier	description
0	0.50868	0.23383	CNhs11760	aorta
1	0.68055	0.46305	CNhs12843	artery
2	0.50006	0.21505	CNhs12856	pulmonic_valve

5. ISM score calculation （Saturation mutagenesis）

For each position, they ran three Saluki forward passes, mutating the reference nucleotide to each of the three possible alternative alleles.

manual: https://github.com/calico/basenji/blob/master/tutorials/sat_mut.ipynb



Code:

```
! ./basenji_sat_bed.py -f data/hg19.ml.fa -l 200 -o output/gata4_sat
--rc -t data/heart_wigs.txt data/params_small.json
../test_models/model_best.h5 data/gata4.bed
```

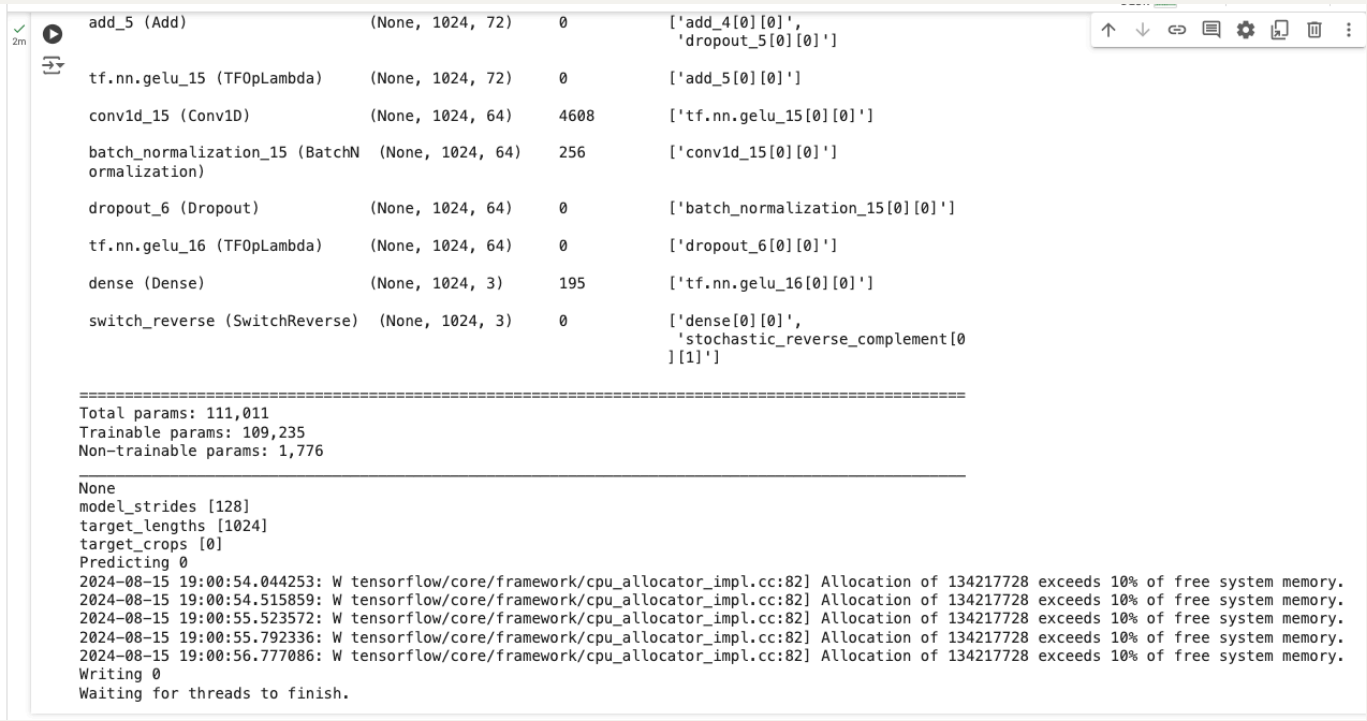
Input:

Option/Argument	Value	Note
-f	data/hg19.ml.fa	Genome FASTA to extract sequences.
-l	200	Saturation mutagenesis region in the center of the given sequence(s)
-o	gata4_sat	Outplot plot directory.
--rc	True	Predict forward and reverse complement versions and average the results.
-t	data/heart_wigs.txt	Target indexes to analyze.
params_file	models/params_small.json	JSON specified parameters to setup the model architecture and optimization parameters.
model_file	models/heart/model_best.h5	Trained saved model parameters.
input_file	data/gata4.bed	BED regions.

gata4 can be download from https://github.com/calico/basenji/raw/master/tutorials/data/gata4.bed

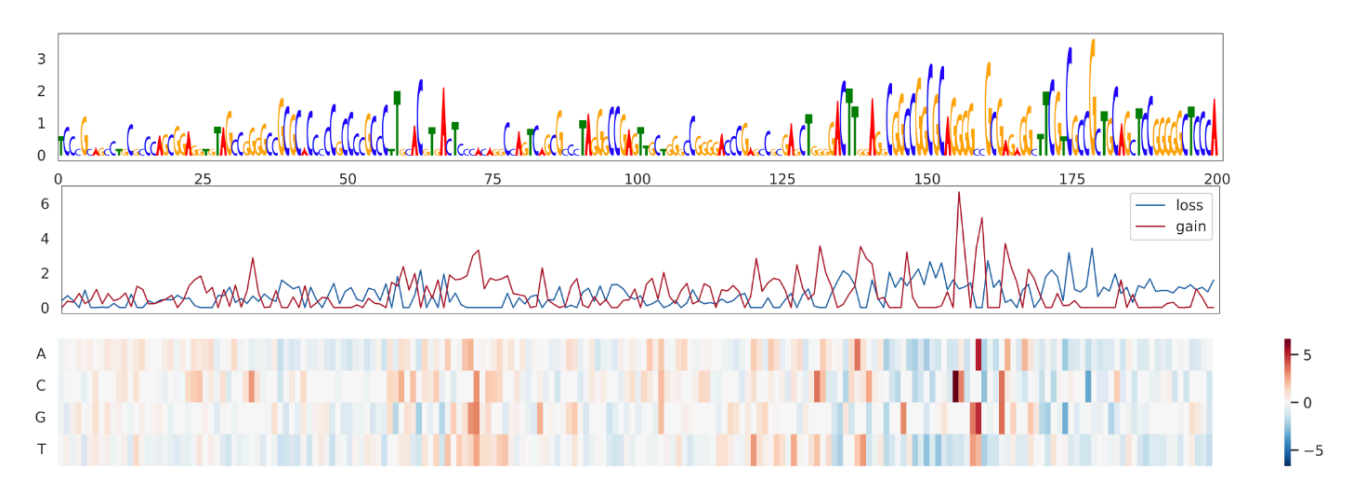
Outout: scores.h5

Process:

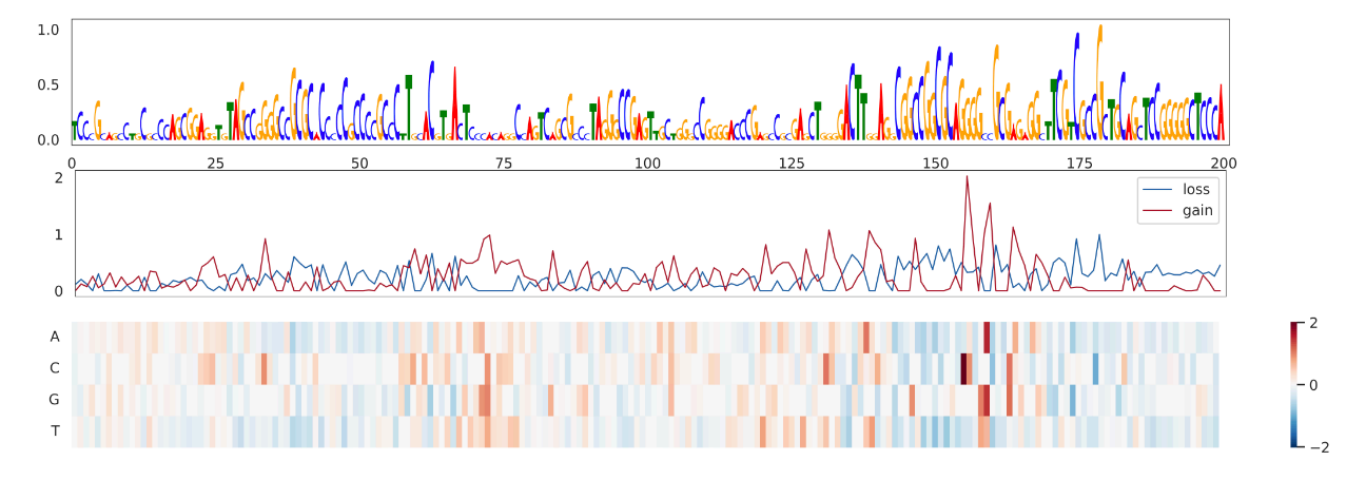


Visualization:

```
! ./basenji_sat_plot.py --png -l 200 -o output/gata4_sat/plots -t
data/heart_wigs.txt output/gata4_sat/scores.h5
```



(aorta)



(artery)